Extracting the Gist of Chinese Judgments of the Supreme Court

Chao-Lin Liu and Kuan-Chun Chen
Department of Computer Science
National Chengchi University
Taipei, Taiwan
void.tw@gmail.com, chaolin@nccu.edu.tw

ABSTRACT

The gist of judgement documents encodes important experience and viewpoints of the Supreme Court, and provides instrumental and educational information for judges, lawyers, practitioners, and students. The Supreme Court in Taiwan appoints senior members to produce the gist for selected judgments of the Supreme Court, but is unable to offer the gist for all judgment documents. Based on our observation of the existing gist statements, we can treat the generation of the gist as a sentence classification problem. We apply machine-learning methods, including gradient boosting, multilayer perceptrons, and deep learning methods with long short-term memory units; and consider legal, linguistic, statistical information, and different word embedding methods to build several classifiers. By using more sophisticated classifiers and more relevant features, we gradually achieved better results, and the best result was 0.9372 in F_1 measure.

CCS CONCEPTS

• Applied Computing → Law • Computing Methodologies → Machine Learning → Machine Learning Approaches →Neural Networks • Computing Methodologies → Machine Learning → Machine Learning Approaches → Classification and Regression Trees • Computing Methodologies → Machine Learning → Machine Learning Algorithms → Ensemble Methods → Boosting • Computing Methodologies → Machine Learning → Learning Paradigms → Supervised Learning → Supervised Learning by Classification

KEYWORDS

Legal Informatics, Information Extraction, Extractive Summarization, Machine Learning, Gradient Boosting, Random Forest, Multilayer Perceptrons, Deep Learning, Long Short-Term Memory (LSTM)

ICAIL'19, June 17–21, *2019, Montréal, Québec, Canada*
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-6754-7/19/06…$15.00
https://doi.org/10.1145/3322640.3326715

ACM Reference format:

Chao-Lin Liu and Kuan-Chun Chen. 2018. Extracting the gist of Chinese judgments of the Supreme Court. In *Proceedings of 2019 International Conference on Artificial Intelligence in Law (ICAIL'19). ACM, New York, NY, USA, 10 pages.* https://doi.org/10.1145/3322640.3326715

1 Introduction

Human experts of the Supreme Court in Taiwan add a gist section to some selected judgment documents of the Supreme Court. These special documents become representative instances for considerations of future judgements and for education of judges and lawyers. The gist section contains statements of the original document that are considered most relevant to the reasons that support the final decisions of the Supreme Court. Most such statements are chosen and copied to the gist sections, but some may be edited slightly.[1]

Although a large proportion of cases finalized at the Supreme Court should be representative and educational, only a small percentage of the judgements were annotated with a gist section due to the constraints of labor costs and the demand in needed time.

We examined the gist sections of the judgement documents of the Supreme Court, and found that a majority of the gist statements were selected from the main body of the documents, though a small proportion was edited manually. Hence, one may treat the problem of producing the gist section as a form of extractive summarization [9], for which task we attempt to find a subset of statements from the original documents to encode the main ideas of the original documents.

We may find many previous research papers on extractive summarization, but the goal of this paper is not doing a complete survey [14]. Hachey and Grover chose sentences to form a summary based on the rhetorical roles of the sentences [11]. Saravanan et al. also considered rhetorical roles to classify sentences to generate summaries with a conditional-random-field

[1] We use "gist" to refer to "裁判要旨" (in Hanyu Pinyin, /cai2 pan4 yao4 zhi3/) in mandarin Chinese. Interested readers who read Chinese may find instances of the gist sections at <http://tps.judicial.gov.tw/faq/index.php?parent_id=757>. There existed extreme exceptions in which the statements of the gist section were creations of the annotator, not extracting from the original statements. (https://law.judicial.gov.tw/FINT/data.aspx?id=B,40,%E5%8F%B0%E4%B8%8A,86,001&ro=0&ty=J1&q=447dc8c3db84cb88f3a8ab8ccacb39e4&sort=DS&) These URL addresses was last accessed on 27 April 2019.

model in [23], and resorted to a legal ontology in another research [24]. Polsley et al. ranked the sentences based on information about word frequency and additional domain-specific knowledge in their demonstration [22]. Yousfi-Monod et al. considered linguistic and statistical features in a naïve-Bayes model for the task [27].

There are relatively fewer ICAIL papers discussing legal proceedings or documents in Chinese. Brown discussed the Chinese tax law [3], Liu et al. studied classification of criminal cases in Taiwan [18], and Wei and Huang analyzed the structures of dialogues in Chinese courts [26].

This paper offers an additional work on the research of legal documents in Chinese. We report methods and their evaluation for algorithmically generating gist statements for documents of Chinese judgments. We utilize linguistic information, statistical information, legal information, and word vectors as features for classifiers that employ technologies of artificial intelligence, including gradient bosting, multilayer perceptrons, and deep learning. Gradually, we produced better and better gist statements in experiments, achieving F_1 measure from about 0.35 to 0.93.

We define our work and summarize the main results in Section 2, present the source corpus in Section 3, discuss the features for our classifiers in Section 4. In Section 5, we explain the settings for our classifiers, go through a sequence of experiments and their results in Sections 6 and 7, and made some concluding remarks in Section 8.

2 Problem Definitions and Main Results

For the judgment documents of the Supreme Court in Taiwan, some are annotated with a section of gist statements. Most of the gist statements were directly selected from the *Reasoning* section [2] in the documents, but some may be selected and edited to become part of the gist. The *Reasoning* section of a judgement document contains the main information about the case, including descriptions about the activities of the accused subjects and about the citations of relevant articles.

Let **R** denote the statements of the *Reasoning* section, our goal is to build a service, **V**, that is able to select some statements **S**, from **R**, that can serve as the gist statements, **G**.

$$\mathbf{V}(\mathbf{R}) \rightarrow \mathbf{S}$$

We treat this task of sentence selection as a classification problem. In addition to considering some relevant features based on knowledge and observations of linguistic and legal perspectives, we also employ different types of word vectors. We realized the service **V** with classifiers that were designed based the concepts of gradient boosting, multi-layer perceptrons, recurrent neural networks, and some ensemble methods of our own.

[2] We use "*Reasoning* section" to translate "理由" (/li3 you2/) in the judgement documents.

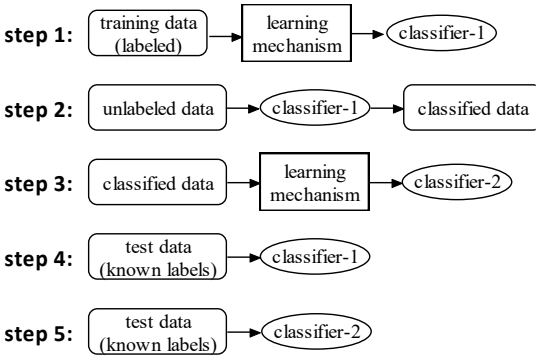

Figure 1: A procedure to be discussed in Section 7

We evaluate the performance of **V** by standard definitions of the precision rate, recall rate, and F_1 measure that are calculated based on the how well the selected statements **S** match the gist statements **G** that were selected and created by human experts.

Different combinations of features and classifiers achieved different quality of the selection. In our experiments, a naively designed classifier achieved a precision rate of only 0.254, a recall rate of 0.570, and F_1 of 0.351. A classifier that was created by a more complex design procedure can accomplish precision of 0.926, recall of 0.949, and F_1 of 0.937.

We also evaluated our classifiers with a practical challenge. Figure 1 shows the procedure of this innovative procedure. Normally, we use a training dataset to obtain a classifier (step 1 in Figure 1). To evaluate the quality of classifier-1, we use the classifier-1 to classify a set of data that we have the correct answers, and compute the precision, recall, and F_1 measure (step 4 in Figure 1).

As an alternative choice, we could use classifier-1 to classify a large set of unlabeled data to produce a set of classified data (step 2 in Figure 1), and check the usability of this classified data. We may do so by using the classified data as a training dataset to train another classifier, classfier-2 in step 3 in Figure 1.

We then use classifier-2 to classify the test data that we already have answers (step 5). If classifier-2 can achieve comparable or even better results than classifier-1 (step 4), then the classified data generated at step 2 can be as useful as the training data in step 1, although we do not check the correctness of their labels manually.

We report results of experiments of this line in Section 7.

3 Target Corpus and Basic Statistics

Since the early 2000CE, the highest judicial administration component of Taiwan, the Judicial Yuan, has started to provide documents about the lawsuits in Taiwan on line. We could

obtain 30554 documents for the judicial precedents and judicial judgments [3] that were annotated with corresponding gist statements. For simplicity of discussion, we will refer to judicial precedents and judicial judgments as judicial judgments henceforth. These documents of the Supreme Court were published for the years between 1927CE and 2017CE. We used these documents in most of the experiments reported in this paper.

We also obtained 276189 documents of judicial judgements of the Supreme Court that were published for the years between 1996 and 2017. These documents do not have their own gist statements, and will be used only in a specific experiment.

We have to convert the online HTML files to our own format. In addition to the *Reasoning* section, the original documents include some fields that are irrelevant to our research, so we preprocessed and converted the files to meet the needs of our experiments.

3.1 Word Level Processing

It is well known that Chinese statements do not use delimiters between words to indicate word boundaries [18][19]. Hence, we have to separate strings of Chinese statements into words with Chinese word segmenters (CWS). We compared two popular CWS, CKIP [4] [19] and Jieba [5], for the current work. CKIP segmenter performed better because it was developed by the Academia Sinica and was trained with more data that were published in Taiwan. Although Jieba segmenter is also very good and convenient, it performs better for Chinese texts that were written in the styles that were closer to those used in Mainland China. For this reason, we chose to use the segmentation results of the CKIP segmenter.

We note that the CKIP segmenter does not identify legal terms very well, but we did not employ a lexicon for Chinese legal terms to improve the segmentation results for two reasons. It was not easy to find a dictionary that would work well for legal documents that were published between 1927 and 2017. Creating and using an ad-hoc internal lexicon would also make our work less reproducible.

Both CKIP and Jieba segmenters annotate the segmented texts with part-of-speech (POS) tags. The POS tags inform us the syntactic information about the words, e.g., whether a word is a verb or a noun [7]. We use the CKIP POS system in our experiments.

3.2 Sentence Level Processing and Analysis

In addition to not having clear word boundaries, a Chinese sentence can be very long (much longer than typical English sentences), uses more complex punctuation marks, and usually corresponds to multiple English sentences. We split Chinese statements by six Chinese punctuation marks: "。", "！", "？", "，", "；", and "："[6], and we call a split part a **sentence segment** or just a **segment**.

Number of judgment documents	30,554
Number of sentence segments	3,142,759
Total number of segments in the gist	362,811
Proportion of gist segments	11.54%
Total number of words in the corpus	27,602,691
Number of distinct words in the corpus	320,302

Table 1: Basic statistics about our corpus

Table 1 shows the basic statistics about our corpus. We obtained more than 3 million segments from 30554 documents, and we have 362811 segments in the gist sections. Overall, 11.54% of the segments were chosen as gist statements. The whole corpus contains more than 27 million words, among which 320302 are distinct.

The gist segments of a judicial judgement normally consist of segments in the *Reasoning* section, in the judicial documents, which are about how the judges reached their decisions. These source segments might be edited before being saved as part of the gist.

A judgement in our corpus, on average, has 102.9 segments, and the standard deviation is 83.51 segments. The average of the percentages of the segments that were chosen into the gist sections of the 30554 judgement is 14.71%,[7] and the standard deviation is 11.31%. Although the judicial system of Taiwan is considered to belong to the civilian law system, the selections of the gist statements were not consistent and must depend on the issues involved in individual judgements [17].[8] It is not easy to include a representative sample judgement and its gist section in an ICAIL paper for discussion because of the numbers and lengths of the relevant statements.[9]

We can apply the **Dice coefficient** [20] to measure the differences between the gist segments and the original segments. The Dice coefficient between two segments, s_1 and s_2, is obtained by dividing the longest common subsequence by the average length of s_1 and s_2. The Dice coefficient is 1.0 if two strings are the same.

Using an English letter to denote a Chinese character, the Dice coefficient between two Chinese strings, ABCDE and ACXE, is

[3] In terms of Chinese, judicial precedents and judicial judgments are "判例" (/pan4 li4/) and "判決" (/pan4 jue2/), respectively.

[4] CKIP CWS available at <http://ckipsvr.iis.sinica.edu.tw/>

[5] Jieba CWS available at <https://github.com/fxsjy/jieba>

[6] They are period, acclamation, question, comma, semicolon, and colon, respectively.

[7] Note that this average is different from 11.54% in Table 1. 11.54% is the percentage of the number of gist statements in all of the statements, while 14.71% is the average of percentages of the gist statements of all individual judgement documents in our corpus.

[8] These statistics are provided to respond to a reviewer's question.

[9] Interested readers may be referred to footnote 1 for information about sample gist statements.

Dice Coefficient	Cumulative Percentage
1.0	77.62%
0.9 or higher	89.04%
0.8 or higher	93.14%
0.7 or higher	95.01%
0.6 or higher	96.28%
0.5 or higher	97.11%
0.4 or higher	97.60%
0.3 or higher	97.95%
0.2 or higher	98.31%
0.1 or higher	98.92%

Table 2: The cumulative distribution of the Dice coefficients between the source and the gist segments

equal to 3/4.5 = 0.667. These two strings share three characters, and their average length is 4.5.

Table 2 shows the cumulative distribution of the Dice coefficients between the original segments and the gist segments. About 77% of the gist segments were copied verbatim from the source segments, and the Dice coefficients between about 89% of the gist statements and their original segments are higher than 0.9.

Based on this observation, when we determine whether a sentence segment that is suggested to be a gist segment matches a real gist segment based on their Dice coefficient. A pair whose Dice coefficient is equal or larger than 0.8 is recorded as a match (Section 2).

Although the gist segments and their original segments are similar, the problem of choosing the right source segments for the gist is not easy. In Table 1, we see that only about 11% (i.e., 362811/3142759) of all of the segments in the *Reasoning* sections were selected as gist segments.

4 Features for the Classification Models

In this paper, we choose sentence segments as gist segments with gradient boosting methods (based on decision trees, referred as the **GB** method henceforth) [6][15] and artificial neural network-based methods (referred as the **ANN** method henceforth) [8]. The GB methods represent a relatively traditional machine-learning approach, and the latter belongs to the school of deep learning. Researchers have shown that both the boosting techniques and neural-network based methods are useful for legal inference [1]. For a candidate sentence segment that may be classified as a gist segment, we consider six types of features. Some of these features can be used for both approaches, and some of them are used only for the GB or the ANN approach, depending on their formats that we will provide more details in Section 5.1.

4.1 Basic Quantitative Features

The first type of features considers basic linguistic information. They include the number of characters, the number of words, and the number of distinct words in a candidate segment.

Based on our observations, the positions of candidate segments in the *Reasoning* section may be related to whether the candidate segment belongs to the gist. Hence, we also recorded the absolute positions of the candidate segments, i.e., the first segment, the second segment, ..., in the *Reasoning* section. Since the lengths of the *Reasoning* sections of different judgment documents may vary a lot, we also recorded the lengths of their *Reasoning* sections, and recorded the relative positions of the candidate segments. Assume that a candidate segment is the x-th sentence segment in a *Reasoning* section of N segments, we would record x, N, and x/N for this segment.

4.2 Features about the Judgement Nature

The nature of the judgments may influence the selections of the gist, and the original documents have included such metadata information.

We consider whether a judgment is for a criminal case or a civil case. We also consider whether a document is recorded as a judgment or as a ruling (which is a category of judicial judgment in Taiwan). In addition, we consider specific words that are used by the courts to indicate subcategories of the judgments and the year the judgment documents were published. (We could show these Chinese words in an oral presentation.[10])

We record the nature of the judgments with one-hot vectors.

4.3 Specific Legal Terms in a Segment

Sometimes, a segment may include specific words that make the segment much more likely to be chosen as a gist segment. These specific words are related to how the argument structure of major premise, minor premise, and conclusions is formulated and presented in a sequence of Chinese statements. In additional, including the identification numbers (including section number and article number) of the cited legal articles[11] may make the segment more likely to be selected as a gist segment. Hence, we record whether a candidate segment contains an identification code as a feature for the candidate segment.

We record whether a segment include these specific terms with one-hot vectors.

4.4 Word Embedding and their Extensions

We employed Gensim[12] to convert words into word vectors so that we can evaluate the effectiveness of deep learning

[10] For instance, "上", "簡上", "抗", and "簡抗".

[11] One may find English translations of Taiwanese laws on line: <https://law.moj.gov.tw/Eng/index.aspx> or <https://db.lawbank.com.tw/Eng/>.

[12] https://radimrehurek.com/gensim/

technologies for gist selection. We considered both CBOW and skip-gram methods [21].

In addition, we also compared the resulting performances of using word2vec and fastText [4]. Similar to using fastText for alphabetic languages, e.g., English, we can obtain a word vector for a Chinese word by combining the vectors of the substrings of the word.

We have two ways to represent the word vectors of the words in a sentence segment. We may directly use a sequence of word vectors, or use the average of the sequence of word vectors of the sentence. We create the vectors for individual documents analogously.

A context for a word can be a sequence of the surrounding words that may provide contextual information about the word. Similar to the creation of sentence and document vectors, we may use a sequence of word vectors, or use the average of the sequence of word vectors of the context. A context for a sentence segment can be a sequence of segments that may provide information about the segment. The vectors for more general contexts can be created analogously.

4.5 Part of Speech

Information about the parts-of-speech of words is important in many tasks of natural language processing, and we consider the POS tags of the words in a sentence segment [7]. An individual POS tag is encoded as a one-hot vector.

For a sentence segment alone, the POS tags for the words in the segment naturally form a sequence, and we convert this POS sequence into the concatenation of the one-hot vectors of first k POS tags in the sequence. While we may choose k arbitrarily, we used k=10 in our experiments, which could cover most Chinese sentences of ordinary length. If a sentence segment has less than k words, we would pad the sequence with zeros.

For a segment in a sequence of segments, we used the average of the POS vectors for the k words in a segment as a feature for the segment.

4.6 Word Embedding of the Opening Words

In Section 4.3 we mentioned that, based on our observation and knowledge about the judgment documents, some words are particular indicative of the possibility for a segment to be selected as a gist statement. The opening words in a sequence of segments are such indicators.

Hence, we apply the word-embedding tool in an innovative way. We extract the opening words of a sequence of sentence segments, compute the word embeddings of the opening words, and use them as features in some of our experiments.

5 Settings for the Experiments

In this section, we provide information about preparing the data for the experiments and setting the parameters for running the classifiers.

5.1 Data Formats

We represent a sentence segment with a set of features when we apply the technique of gradient boosting or the multilayer perceptrons to build the classifiers. Features discussed in Sections 4.1 through 4.3 can be combined into a feature set, for instance.

In contrast, when using the deep learning methods for building classifiers for documents, we convert an individual word into a vector. A sentence segment thus becomes a vector of vectors. In Section 4.4, we mentioned that there are two ways to represent a sentence segment: either directly use the vector of word vectors or use an average of the component word vectors.

For the former choice, the classifier has to handle a sequence of feature to determine the type of the candidate sentence segment. We expect this to be a better choice because we preserve the details about individual word vectors in the sequence, and we use this type of features for LSTM deep learning approaches.

To determine whether a sentence segment should be chosen as a gist statement, we may consider the context of the candidate segment, where the context consists of the surrounding sentence segments of the candidate segment. Hence, in this type of experiments, we also use a sequence of features for a candidate sentence segment.

As a general principle, we use 70% of the data listed in Table 1 as the training set, 10% as the validation set, and 20% as the test set. We conduct the sampling at the sentence-segment level, so the segments of a judgement document may be used for training, validation, or test. The datasets were resampled for every different sets of experiments.

5.2 Setting the Classifiers

We will employ three different types of classifiers. They are the gradient boosting with decision trees (**GB**), traditional multilayer perceptrons (**MLP**, henceforth), and the recurrent neural networks.

The method of gradient boosting with decision trees has performed very well in many machine-learning problems [6], so we chose it as one of our classification methods. We rely on the LightGBM implementation that Ke et al. reported in NIPS 2017 [15]. When running the LightGBM, we used the *log loss* function [2] for the objective function, and chose *leaf-wise* as the tree growth algorithm. We set *max leaves* to 2047, *learning rate* to 0.05, and *earlier stopping patient* to 50.

Classifiers	Attributes	Precision	Recall	F_1
GB	A1	0.2609	0.5836	0.3606
GB	A1+A2	0.4769	0.6327	0.5438
GB	A1	0.2616	0.5878	0.3621
GB	A1+A3	0.3437	0.5557	0.4247
5-layer MLP	A1	0.2537	0.5699	0.3511
5-layer MLP	A1+A2	0.2973	0.5680	0.3903
5-layer MLP	A1	0.2462	0.6293	0.3539
5-layer MLP	A1+A3	0.3475	0.5188	0.4162

Table 3: Effects of using some base feature sets

We employed Keras[13] to realize the classification models that include multilayer perceptrons and recurrent neural networks. The feedforward neural networks are fully connected. We used both the long short-term memory (**LSTM**, henceforth) units and bidirectional LSTM units (**BiLSTM**, henceforth) in our experiments [13][25]. These LSTM units offer capability of learning the contextual information in texts, so are expected to offer better performance.

We chose to use five layers of MLP or LSTM in our experiments. When adding more layers in a certain explorative experiments, we may observe good performance, but we may also encounter the vanishing gradient problems, so we used five layers in current experiments.

When training the LSTM units, we used the Adam method for optimization [16], the *log loss* function as the objective function, 100 as the *batch size*, and 2 for the earlier stopping patient. The size of word vectors is **200**. We consider **10** words surrounding a word when generating the word vector for the word. When creating a context for a candidate segment, we considered **15** segments surrounding the candidate segment.

In experiments that use the ANN-based methods, the final outputs of the neurons are in the range of [0, 1], where 0 means very unlikely for a candidate sentence segment to be selected as a gist segment and 1 means very likely. We rely on a set of validation data to dynamically choose the threshold values for whether or not to accept a candidate segment as a gist statement.

6 Empirical Evaluations

We report results of some empirical evaluations in this section. We check the effectiveness of using individual features, discuss the contribution of contextual information, integrate the features to achieve better results, and finally try to further improve the results by building ensemble models.

6.1 Basic Comparisons

We could use different sets of features with the classifiers to observe the effectiveness of the different combinations. Table 3 shows the observed performances of four paired experiments. The leftmost column lists the classifiers, the "Attributes" column shows the sets of features that were used in the experiments, and the remaining columns show the precision rate, the recall rate, and the F_1 measure, respectively. A1, A2, and A3 refer to features that we discussed in Sections 4.1, 4.2, and 4.3, respectively.

There are two rows for one paired experiment. For instance, the second row of Table 3 shows the results of running a GB model with attribute A1, and the third row shows the results of running a GB model with A1 and A2.

We resampled and split our data into training and test sets for each experiment, so we observed different performances even when the combination of the classifier and the attributes were the same. For instance, the statistics listed in the second and the fourth rows were not the same. We note that the training and test sets for experiments in a paired experiment were the same, for otherwise the comparison would be meaningless.

The statistics in Table 3 indicate that adding more attributes to A1 made our classifiers more effective. It is interesting to notice that adding A2 to A1 is more helpful than adding A3 for the GB model, but adding A3 is more helpful for the 5-layer NN model.

The statistics in Table 3 also show that the results were not satisfactory, and have large room for improvement. Using A1 and A2 with a GB model achieved the best performance in Table 3 (F_1 is equal to 0.5438 in the third row).

Classifiers	Attributes	Precision	Recall	F_1
GB	A1+A2+A3+A6	0.5511	0.6471	0.5953
GB	A1+A2+A3+ A5+A6	0.5888	0.6599	0.6223
5-layer MLP	A1+A2+A3+A6	0.4586	0.5944	0.5178
5-layer MLP	A1+A2+A3+ A5+A6	0.5010	0.5817	0.5383

Table 4: Effects of using more base feature sets

Along this line of intuition, we added A5 and A6 in the experiments, and Table 4 lists the observed results. A5 and A6 refer to features that we discussed in Sections 4.5 and 4.6, respectively. Features that provide domain-dependent information were helpful for achieving better results, but the overall results are still not satisfactory.

6.2 Word Embeddings with LSTM

[13] https://keras.io/

		Size	Precision	Recall	F_1
word2vec	CBOW	50	0.8694	0.8885	0.8788
		100	0.8847	0.8968	0.8907
		200	0.8908	0.9118	0.9012
		300	0.8908	0.9122	0.9014
	skip-gram	50	0.8896	0.9087	0.8991
		100	0.8820	0.9110	0.8963
		200	0.8967	0.9216	0.9090
		300	0.9059	0.9289	0.9172
fastText	CBOW	50	0.8734	0.8847	0.8790
		100	0.8796	0.9041	0.8917
		200	0.8785	0.9078	0.8929
		300	0.8850	0.9110	0.8978
	skip-gram	50	0.8795	0.8899	0.8847
		100	0.8996	0.9102	0.9049
		200	0.8952	0.9172	0.9060
		300	0.9020	0.9269	0.9143

Table 5: Effects of using word embeddings with LSTM units

We tried different ways of creating word embeddings and using the word embeddings to build embeddings for sentence segments. First, we considered 10 words around an individual word when we created vectors for the word. We created the embedding for a sentence segment by averaging the word embeddings of the words in a sentence segment. We used an LSTM model, and considered 15 sentence segments before and after the candidate sentence segment that we would determine whether to choose it as a gist segment. The sentence embeddings of the surrounding segments for the candidate segment provided contextual information for our decisions.

Table 5 shows the observed results. The leftmost two columns indicate the types of word embedding operations. The "Size" column records the sizes of word vectors.

The statistics in Table 5 did not show appreciable differences between using word2vec and fastText. Using skip-gram led to slightly better results than using CBOW. Converting words to longer word vectors usually awarded us with a bit better results, but the differences are not very significant, while we need more memory space for the experiments. Hence, we set the sizes of word vectors to 200 in other experiments.

Using the word embeddings with the LSTM units clearly achieved better results (Table 5) than when we classified the sentence segments with the GB and MLP models based only on the basic features (Tables 3 and 4).

6.3 Contextual Effects and Domain Knowledge

We evaluated the contribution of POS information in more experiments. We compared two ways of using the POS information. In Section 4.5, we create a one-hot vector of POS labels for each individual word in a candidate sentence segment (CSS). Since CSS contains multiple words, a CSS can be represented as a sequence of one-hot vectors for POS labels. The second row in Table 6 shows the results of this approach.

classifier	features	Precision	Recall	F_1
5-layer LSTM	POS vectors for a CSS	0.3044	0.5552	0.3932
	Contextual POS vectors for a CSS	0.8675	0.8826	0.8750
	Contextual opening-word embeddings	0.8658	0.9031	0.8840

Table 6: Effects of alternative ways of using POS information

Alternatively, we calculated the average of the one-hot vectors for the words in a sentence segment, and used the average vector to represent a segment. When classifying a CSS, in addition to using the average vector for the candidate segment, we considered the average vectors of the 15 previous and 15 following segments as the contextual information. The third row of Table 6 shows the observed results for this approach.

We used POS information in both experiments, but we obtained quite different results in the second and the third rows of Table 6. The POS information about the candidate sentence segment alone did not appear to be effective, as the statistics in the second row suggest. When using the POS information about the context, we achieved the statistics in the third row, and the results were much better, although the results were not as impressive as those shown in Table 5.

As another support for the effectiveness of contextual information, we used the word embeddings of the opening words in a sequence of sentence segments as the features for classifying a candidate sentence segment (Section 4.6). The last row of Table 6 lists the observed results. Considering the context of only the opening words in the LSTM model could achieve 0.8840 in the F_1 measure.

Although we expected that some words are particularly indicative for a sentence segment to be selected as a gist statement (Sections 4.3 and 4.6), the results in the last row are still surprising. The results were not as good as those listed in Table 5, but were comparable with those that we observed when using the average vectors of POS tags.

6.4 Integrated Experiments

We used all of the features which we explained in Section 4 in the experiments reported in this subsection. Features used in these experiments differed in whether they include contextual information. Although possible, we did not include the features defined in Sections 4.1 to 4.3 as contextual information, so they were not make "sequential" as we explained in Section 5.1. Features explained in Sections 4.3 to 4.6 can be sequential or non-sequential features. The average vector of the candidate sentence segment was used to represent the segment itself, and the average vectors of each of the surrounding segments were included in the features to form a sequence of features.

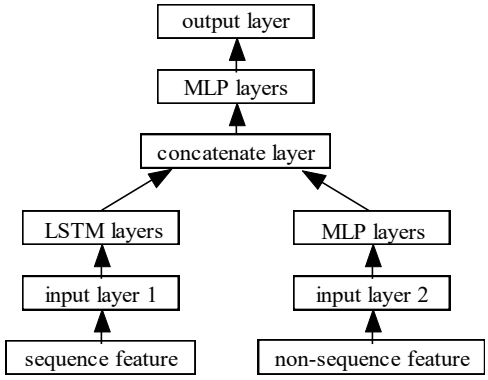

Figure 2: the network structure for integrated experiments

classifier	features	Precision	Recall	F_1
RF	**non-sequence**	0.5160	0.6161	0.5616
MLP	**non-sequence**	0.5664	0.6564	0.6081
GB	**non-sequence**	0.8443	0.8569	0.8506
5-layer BiLSTM	**sequence**	0.9008	0.9122	0.9065
Mixed	**All**	0.9142	0.9291	0.9216

Table 7: Effects of alternative types of classifiers

Figure 2 provides the skeleton for the network structure that we used to combine the sequential and the non-sequential features for the classifiers. We refer to this structure as a **Mixed** model. The non-sequential part does not include information about the surrounding sentence segments, so needs not to be processed by the LSTM layers. The intermediate results for the sequential and non-sequential features were concatenated and fed to the final MLP layers to produce the final prediction.

Table 7 lists the observed results of using random forest (RF)[12], GB, MLP, bidirectional LSTM units (BiLSTM). From top to bottom, we list the classifiers in an order of increasing modernity, and we achieved gradually better results alone the way.

6.5 Ensemble Models

We built three basic classifiers that respectively use the GB model, 5-layer LSTM units, and the Mixed model methods as the level-1 classifiers. The outputs of the level-1 classifiers are in the range of [0,1], where 0 means very unlikely for a candidate sentence segment to be selected as a gist segment and 1 means very likely. We used three different ways to build the level-2 classifiers to combine the outputs of the level-1 classifiers.

The simplest method to integrate the level-1 classifiers is to use the **arithmetic average** of the outputs of the basic classifiers as the basis to determine the level-2 classification. We can also train a model that find the weights for the outputs of the level-1 classifiers, and then use the **weighted sum** to determine the level-2 classification. The third method is to train a **model selector** that learns to choose the most reliable level-1 classifier given the inputs of the level-1 classifiers. We implemented the model selector as a random forest.

classifier	Precision	Recall	F_1
GB	0.8420	0.8674	0.8545
5-layer LSTM	0.8930	0.9262	0.9093
Mixed	0.9102	0.9171	0.9136
arithmetic average	0.9258	0.9488	0.9372
weighed sum	0.9281	0.9456	0.9372
model selector	0.9072	0.9233	0.9152

Table 8: Performance of ensemble models

Table 8 lists the results. The second to the fourth rows show the performance of the level-1 classifiers. Recall that we resampled our data for training and testing for each different sets of experiments, so the performances of the classifiers that used the same underlying technologies may not achieve exactly the same performance in different sets of experiments, e.g., the performance of the LGBM in Tables 7 and 8 are similar but different.

The fifth to the seventh rows of Table 8 show the performances of the ensemble models. Overall, the results of the level-2 classifiers are better than those of the level-1 classifiers. Although it is very simple to implement the arithmetic average, its performance is relatively good.

7 Using Machine-Generated Data for Training

Although we have collected all of the published documents (before 2017) of the Supreme Court that have been annotated with gist statements, the total number of such documents is not huge (30554 in Table 1). Since we have achieved high F_1 measures in our experiments, we could gather more documents of the Supreme Court that do not contain gist sections, and use our classifiers to select gist statements for such documents.

We processed these new documents, and produced more than two million sentence segments that were useful for training a new classifier. The problem is that it was very difficult for us to examine the correctness of the classification results of these two million sentence segments in person.

As an explorative attempt, we pretended that these sentence segments were classified by human experts, and used them as part of our training data in additional experiments. We would observe the effects of adding the extra training data on the final classification results.

We have outlined the plan for this exploration in Figure 1 in Section 2. At the first stage of this three-stage exploration, we had 2199932 and 314275 sentence segments in documents that

classifier	Precision	Recall	F_1
first-stage	0.5980	0.7128	0.6504
second-stage	0.6033	0.6930	0.6450
third-stage	0.4225	0.7343	0.5367

Table 9: Algorithmically classified data proved useful

have gist statements as training and validation sets, respectively. We put aside test documents that are also annotated with gist statements, and these test documents include 628552 sentence segments. We trained a classifier using the word embeddings (Section 4.4) as the feature with a 5-layer LSTM model.

The "first-stage" row of Table 9 shows the quality of classification results of this classifier. Note that the statistics are much worse than those listed in correspond rows for 5-layer LSTM in Tables 7 and 8. The differences are consequences of our downsizing our settings for the experiments.

In this set of experiments, we considered only **two** surrounding sentence segments as the context (not 15 segments as we stated in the last paragraph of Section 5.2) for a candidate segment. We had to do so, for otherwise our computer disk could not accommodate the training data for the second stage, which we explained next.

We then used the first-stage classifier to classify sentence segments of documents that were not selected to be annotated with gist statements. As the results of our operations, these documents would have gist statements, and we added 2080954 new segments (with our own classifications) into the training data of the first stage to train another classifier (also using a 5-layer LSTM model), and employed this new classifier to classify the same test data for the first stage, i.e., 628552 sentence segments.

After adding 2080954 new segments into our training data, we had 2199932 plus 2080954 segments in the training data. We needed much more disk space to handle these 4280886 sentence segments if we still wanted to consider 15 surrounding segments for providing contextual information, which we could not afford. Hence, we reduced the number of surrounding segments from 15 to two in this whole set of experiments.

The second and the third rows of Table 9, respectively, show the results of the first and the second stage experiments, suggesting that adding more training data did not affect the classification results for the test data significantly. The F_1 measures are very close.

At the third stage, we changed the training data for the second stage. We removed the documents that were annotated by human experts, used only the sentence segments that were labelled by our classifiers (created at the first stage) to train a third classifier, and used this last classifier to classify the test segments (still those 628552 test segments).

The "third-stage" row in Table 9 shows the experimental results. Although the results were not as good as those listed in the "first-stage" and the "second-stage" rows, we thought the results are quite encouraging. We trained the third classifier using only the data that were algorithmically labeled by the first-stage classifier. We did not check or change the output of the first-stage classifier in any way. The classification results of the third-stage classifier suggest that our system may be practically useful for annotating the gist statements for legal documents.

Of course, we must admit that further experiments on more powerful computing environments are necessary to validate this promising observation. In particular, we should try to expand the amount of surrounding sentence segments that are used in the current exploration.

8 Concluding Remarks

Having achieved F_1 of as high as 0.9372 (Table 8) is a convincing indicator that machine-learning methods offer a practically useful way for assisting the selection of the gist statements for judgment documents of the Supreme Court in Taiwan. The experimental results reported in Section 7 further strengthen the general applicability of the algorithmically generated gist statements, and show us the possible way of creating large legal database [10] with the support of artificial intelligence technologies.

The experiments reported in Section 6 provide supportive evidence for using LSTM units for the gist selection problem. The experiments reported in Section 6.3 provide evidence for the importance of contextual information. LSTM units offer a natural mechanism to take advantage of the contextual information, and the deep networks grant us the flexibility to combine the basic features to form high-level features that helped us accomplish better results than using the random forests, gradient boost, or the traditional MLPs.

We have not completed a large-scale analysis of the classification errors yet. In Section 5.1, We stated that we did the classification for individual candidate sentences, so a sentence segment in a judgement document may be used either for training, for validation, or for test. Splitting data by individual segments helped our systems learned something from each judgement, so is helpful for achieving better results.

We just obtained 19 judgement documents of 2018CE that have gist sections, and used our LSTM classifier to select the gist sentences. The classifier achieved F_1 that are better than 0.8 for four of these documents, and achieved F_1 that are better than 0.6 for nine documents. However, the classifier also failed completely for another four documents. It is hard to offer a scientific explanation for the observed performance of the LSTM-based classifiers, partially because of the limited amount of test instances and partially because of the nature of the deep learning mechanism. The main issues involved in those judgements must matter, and our classifier may not have

obtained sufficient training data to learn the knowledge to achieve very good results for these unforeseen instances.

ACKNOWLEDGMENTS

The research was supported in part by contracts MOST-104-2221-E-004-005-MY3 and MOST-107-2221-E-004-009-MY3 of the Ministry of Science and Technology of Taiwan. Experiments reported in this paper were converted from the Master's thesis of K.-C. Chen [5] whose graduate studies were under the supervision of C.-L. Liu who translated part of the thesis for writing this paper.

REFERENCES

[1] E. Alfaro, N. Garcia, M. Gamez, and D. Elizondo (2008). Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks. *Decision Support Systems*, 45, 110–122.

[2] E. Alpaydin (2010). *Introduction to Machine Learning*, the MIT Press.

[3] G. Brown (2003). CHINATAX: exploring isomorphism with Chinese law. *Proceedings of the Fourth International Conference on Artificial Intelligence and Law*, 175–179.

[4] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5, 135–146.

[5] K.-C. Chen (2018). *Automatic Extraction of Gist of Chinese Judgments of the Supreme Court*, Master's Thesis, National Chengchi University, Taiwan. (advisor: Chao-Lin Liu)

[6] T. Chen and C. Guestrin (2016). XGBoost: A scalable tree boosting system. *Proceedings of the Twenty-Second SIGKDD Conference on Knowledge Discovery and Data Mining*, 785–794.

[7] T. Gonçalves and P. Quaresma (2005). Is linguistic information relevant for the classification of legal texts? *Proceedings of the Tenth International Conference on Artificial Intelligence and Law*, 168–176.

[8] I. Goodfellow, Y. Bengio, and A. Courville (2016). *Deep Learning*, the MIT Press.

[9] C. Grover, B. Hachey, I. Hughson, and C. Korycinski (2003). Automatic summarisation of legal documents. *Proceedings of the Ninth International Conference on Artificial Intelligence and Law*, 243–251.

[10] A. Gupta, A. Z. Wang, K. Lin, H. Hong, H. Sun, B. L. Liebman, R. E. Stern, S. Dasgupta, and M. E. Roberts (2017). Toward building a legal knowledge-base of Chinese judicial documents for large-scale analytics. *Proceedings of the Thirtieth Conference on Legal Knowledge and Information Systems*, 135–144.

[11] B. Hachey and C. Grover (2006). Extractive summarization of legal texts. *Artificial Intelligence and Law*, 14(4), 305–345.

[12] T. K. Ho (1995). Random decision forests. *Proceedings of the Third International Conference on Document Analysis and Recognition*, volume 1, 278–282.

[13] S. Hochreiter and Jürgen Schmidhuber (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.

[14] A. Kanapala, S. Pal, and R. Pamula (2017). Text summarization from legal documents: a survey. *Artificial Intelligence Review*, 1–32, Springer.

[15] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems 30*, 3149–3157.

[16] D. P. Kingma and J. Ba (2015). Adam: A method for stochastic optimization. *Proceedings of the Third International Conference for Learning Representations*, 2015.

[17] H.-L. Lin (2010). *A Study on the Content Analysis and Metadata Design of Court Decisions*, Section 4.1, Master's Thesis, National Chiao-Tung University, Taiwan. (advisor: Hao-Ren Ke)

[18] C.-L. Liu, C.-T. Chang, and J.-H. Ho (2003). Classification and clustering for case-based criminal summary judgments. *Proceedings of the Ninth International Conference on Artificial Intelligence and Law*, 252–261.

[19] W.-Y. Ma, and K.-J. Chen (2003). Introduction to CKIP Chinese word segmentation system for the first international Chinese word segmentation bakeoff. *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*, 168–171.

[20] C. D. Manning and H. Schütze (1999). *Foundations of Statistical Natural Language Processing*, the MIT Press.

[21] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean (2013). Distributed Representations of Words and Phrases and their Compositionality. *Proceedings of the Twenty-Sixth International Conference on Neural Information Processing Systems*, volume 2, 3111–3119.

[22] S. Polsley, P. Jhunjhunwala, and R Huang (2016). CaseSummarizer: A system for automated summarization of legal texts. *Proceedings of the Twenty-Sixth International Conference on Computational Linguistics: System Demonstrations*, 258–262.

[23] M. Saravanan, B. Ravindran, and S. Raman (2006). Improving Legal Document Summarization Using Graphical Models. *Proceedings of the Nineteenth Conference on Legal Knowledge and Information Systems*, 51–60.

[24] M. Saravanan, B. Ravindran, and S. Raman (2007). Using legal ontology for query enhancement in generating a document summary. *Proceedings of the Twentieth Conference on Legal Knowledge and Information Systems*, 171–172.

[25] M. Schuster and K. K. Paliwal (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11), 2673–2681.

[26] B. Wei and J. Huang (2015). Modelling dialogues in court using a gradual argumentation model: A case study. *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Law*, 138–147.

[27] M. Yousfi-Monod, A. Farzindar, and G. Lapalme (2010). Supervised machine learning for summarizing legal documents. *Proceedings of the Twenty-Third Canadian conference on Advances in Artificial Intelligence*, 51–62.