Deep Learning-Based Analysis of Judicial Decision Bias: A Postprint

Authors: Wang Yepei, Song Mengjiao, Wang Xuan, Zhao Zhihong

Date: 2018-05-20T00:00:00+00:00

Abstract

The tendency analysis of judgment results in judicial documents refers to determining whether the judgment outcomes support the plaintiff' s litigation requests. The tendency analysis of judgment results holds significant importance for subsequent tasks such as the standardization of judicial documents and the recommendation of litigation attorneys; however, effective analysis models are currently lacking. To leverage the vast amounts of judicial document data, this paper proposes a model for the tendency analysis of judgment results. Key features are extracted from semi-structured judicial documents, multiple entities in the judgment results are identified and cleaned using fuzzy matching, and the processed data is then fed into an LSTM-based deep learning neural network for tendency determination. Experimental results on datasets from three types of causes of action demonstrate that the model achieves an accuracy of up to 98.3%, validating its high effectiveness in the task of tendency analysis of judgment results.

Full Text

Abstract

The orientation analysis of judgment results in judicial documents refers to determining whether the judgment supports the plaintiff' s claims. This analysis is crucial for subsequent tasks such as standardizing judgment documents and recommending litigation lawyers, yet effective analysis models remain lacking. To fully leverage the massive repository of judgment documents, this paper proposes a model for orientation analysis of judgment results. The model extracts key features from semi-structured judgment documents, identifies and cleans multiple entities in judgment results using fuzzy matching, and processes the results through a deep learning neural network based on LSTM for orientation determination. Experiments on datasets from three types of cases demonstrate

that the model achieves a maximum accuracy of 98.3%, verifying its high effectiveness in orientation analysis of judgment results.

Keywords: deep learning; LSTM; judgment results; orientation analysis

0 Introduction

Judicial documents represent the authoritative written conclusions issued by courts exercising judicial power over substantive or procedural matters in specific cases. These documents record the adjudication process, basis, reasoning, and outcomes in written form [1]. The orientation analysis of judgment results in judicial documents involves analyzing the judgment to determine whether it supports the plaintiff' s claims. This analysis can be used to evaluate the appropriateness of terminology usage in judgment documents and to analyze litigation lawyers'win rates, playing a significant role in applications such as judgment document standardization and litigation lawyer recommendation. However, judgment documents are semi-structured texts, and issues such as non-standardized content, human recording errors, and inconsistent terminology usage pose substantial challenges to orientation analysis.

The orientation analysis of judgment results resembles short-text sentiment analysis problems. First, both tasks deal with short texts, typically not exceeding 200 characters. Second, while sentiment analysis primarily involves extracting sentiment information units and analyzing their orientation, judgment result orientation analysis involves extracting judgment information units and analyzing their orientation. Therefore, methods similar to short-text sentiment analysis can be employed to solve judgment result orientation analysis problems.

Common approaches for short-text sentiment analysis include dictionary-based rule methods and machine learning-based methods. Due to the difficulty of dictionary-based methods in adapting and generalizing across different types or topics of corpora, along with their heavy reliance on expert domain knowledge, they have gradually been replaced by or integrated with machine learning methods in recent years. Among these, deep learning methods require no prior knowledge and utilize deep neural networks to learn features from word vectors to generate language models.

Unlike short-text sentiment analysis, judgment results are embedded within semi-structured texts like judgment documents and cannot be obtained directly. Moreover, since the object entities in judgment results have a decisive impact on the orientation outcome, and judgment results often use personal names rather than unified legal entities, accurately identifying multiple entities in the judgment and cleaning them becomes necessary. No prior work has applied machine learning methods, including deep learning, to judgment result orientation analysis.

To address these challenges, this paper extracts key features from judgment documents, identifies multiple entities in judgment results using fuzzy match-

ing, and designs an orientation analysis model based on deep learning methods using Long Short-Term Memory (LSTM) networks. Using only a small amount of annotated judgment results as training and test sets, experiments compare the accuracy of dictionary-based rule methods and deep learning methods for orientation analysis, verifying the effectiveness of the proposed approach. The main contributions include: (a) applying deep learning to judgment result orientation analysis and verifying its effectiveness; (b) investigating the impact of neural network depth on classification accuracy in this task; (c) utilizing fuzzy matching to solve the problem of multiple entity recognition in judgment results; and (d) designing an experimental model tailored to the linguistic characteristics of judgment documents, including text preprocessing, that requires no manual intervention for intermediate steps after training—simply inputting a judgment document yields the result label.

1 Related Work on Sentiment Analysis

In recent years, sentiment analysis, also known as polarity analysis, has become one of the most popular research topics in NLP. Research methods have evolved from initial dictionary-based rule approaches to machine learning-based approaches.

Dictionary-based rule methods typically require constructing a sentiment dictionary first, then calculating the sentiment of the entire text based on the prior sentiment of sentiment words in the test text within the dictionary. Problems with this approach include: (a) dictionaries cannot cover all sentiment vocabulary, particularly rapidly evolving online vocabulary; and (b) sentiment words themselves may have multiple meanings in different contexts. To address the first issue, Turney et al. [2] proposed determining the sentiment polarity of the entire text based on the correlation between words in the test text and words in a seed dictionary; Mohammad et al. [3] attempted to generate dictionaries suitable for social media sentiment analysis. For the second issue, Jijkoun et al. [4] proposed generating topic-specific sentiment dictionaries to reduce semantic diversity caused by topic divergence.

Machine learning methods were first applied to sentiment analysis by Pang [5] in 2002. These methods typically transform sentiment analysis into a pattern classification problem, building classification models to predict sentiment polarity. When building models, pre-annotated data is required. References [6,7] utilized traditional machine learning methods, including Support Vector Machines and Naive Bayes, combining different classifiers to improve accuracy. References [8,9] attempted to combine dictionary-based rules with traditional machine learning, achieving promising results.

In recent years, various deep learning models centered on Recurrent Neural Networks (RNN) [10] have been applied to sentiment analysis tasks, achieving good results. These include the LSTM model proposed by Hochreiter et al. [11], the GRU model [12] that reduces computational complexity, and Bidirectional

LSTM [13] that can mine more contextual information. Currently, most research focuses on English texts, while Chinese research primarily targets Weibo sentiment analysis. Tang et al. [14] proposed using word embedding to represent word information; Vo et al. [15] used distributed word representation and deep learning feature extraction methods, achieving good classification results. Teng et al. [16] proposed fusing LSTM models with topics; Liang et al. [17] combined LSTM with polarity shift models; and Zhang [18] designed an attention-based LSTM model. However, no prior work has applied machine learning methods, including deep learning, to judgment result orientation analysis.

2 Deep Learning-Based Orientation Analysis Model for Judgment Results

The deep learning-based orientation analysis model for judgment results designed in this paper is shown in [Figure 1: see original paper].

2.1 Judgment Document Preprocessing

Judgment documents are semi-structured texts, typically structured as shown in [Figure 2: see original paper]. They contain much content unrelated to orientation analysis, such as adjudication processes, reasoning, and basis. Additionally, since judgment results frequently use personal names, company names, and other appellations rather than legal terms like "plaintiff" and "defendant," these long company or institution names would be split into multiple words during segmentation, causing information loss and ultimately affecting deep training effectiveness. Therefore, before segmentation, the names of plaintiffs and defendants are extracted from judgment documents and replaced with legal terms in the judgment results.

The preprocessing of judgment documents mainly consists of the following steps:

a) **Data Extraction**: Extract key features such as plaintiffs, defendants, and judgment results from judgment documents. Due to the semi-structured nature of judgment documents, extracting paragraphs containing key features is relatively easy, but extracting accurate features from paragraphs requires designing different regular matching conditions based on feature context.

b) **Data Cleaning**: Use fuzzy matching to identify personal names, company names, and other appellations in judgment results, replacing them with corresponding legal terms such as "plaintiff" and "defendant." In this step, some company names in judgment results are not completely consistent with those extracted from plaintiffs and defendants. For example, the extracted plaintiff name might be "Beijing ** Engineering Technology Co., Ltd.,"while the judgment result uses "** Engineering Technology Co., Ltd." These appellations are typically substrings of the full name. Therefore, the longest common substring algorithm is used for fuzzy matching in the

final data cleaning process. Let W be the set of all plaintiff and defendant names, s_k be the longest common substring between the k-th name w_k and the judgment result, and r_k be the length ratio between the k-th longest common substring s_k and the k-th name w_k.

$$W = \{w_1, w_2, \cdots, w_n\}$$

$$r_k = \frac{length(s_k)}{length(w_k)}$$

The identity ("plaintiff" or "defendant") corresponding to the maximum value in the set $\{r_k\}$ is selected to replace the longest common substring in the judgment result.

c) **Data Annotation**: The judgment results obtained from the previous step are manually annotated as either "support plaintiff" or "not support plaintiff." For special cases, additional rules were formulated as shown in .

Table 1: Special Case Annotation Rules - In a single judgment result, if the plaintiff is partially supported, it is judged as supporting the plaintiff. Example: "The defendant shall return the plaintiff' s project deposit of 1,376,505 yuan within ten days from the effective date of this judgment, and the plaintiff' s remaining litigation is dismissed."- Withdrawal of lawsuit is judged as supporting the plaintiff. Example 1: "The plaintiff' s withdrawal is permitted." Example 2: "This case is treated as the plaintiff' s withdrawal of the lawsuit." - Dismissal of the defendant' s counterclaim is judged as supporting the plaintiff; similarly, dismissal of the plaintiff' s counterclaim is judged as supporting the defendant. Example: "The defendant' s counterclaim against the plaintiff is not accepted."

During annotation, three people performed manual labeling, and the final label for each judgment result was determined by comprehensively considering the three annotations to reduce the possibility of human error.

d) **Segmentation**: The judgment results that have completed the above steps are segmented as input for the next stage.

2.2 Text Vectorization

Text vectorization represents segmentation results using numerical vectors. There are generally two methods: One-hot Representation and Distributed Representation. This paper adopts Distributed Representation because it uses low-dimensional vectors to represent each word and, by distributing word meanings across different dimensions, enables similarity judgment between words based on vectors.

Although some open-source word vectors exist, the training corpora differ significantly from the judgment document scenarios required in this paper. Therefore,

to fully utilize judgment document content, a new word vector was generated for text vectorization. The steps are as follows:

a) Segment the judgment documents;
b) Train to obtain word vectors;
c) Represent the segmentation results from the previous stage using word vectors.

2.3 Deep Neural Network

[Figure 3: see original paper] shows the deep neural network model designed in this paper, which takes word vector-represented segmentation results as input to the LSTM network. Since the final output of orientation analysis is a classification label, only the output result of the last LSTM unit needs to be considered. As the output result is a vector, an additional hidden layer is added for feature selection, with the final output label obtained using a sigmoid activation function.

The deep neural network was trained to completion to obtain the final model. This paper designed single-layer LSTM, two-layer LSTM, and three-layer LSTM architectures to analyze the impact of neural network depth on judgment result orientation analysis. The LSTM layers marked with dashed lines in [Figure 3: see original paper] indicate that the number of LSTM layers varies across different experiments.

This paper uses the LSTM model proposed by Hochreiter et al. as the core because, for RNNs, Bengio et al. [19] discovered that RNNs suffer from the vanishing gradient problem, causing later time nodes to have diminishing perception of earlier time nodes. LSTM adds two concepts to the RNN: cell state and gates. The cell state is transmitted throughout the entire LSTM hidden layer, and information stored within it is not lost but can be added to or deleted through different gates. Gate structures are used to select information, and LSTM contains three types of gates: forget gate, input gate, and output gate, as shown in [Figure 4: see original paper].

[Figure 4: see original paper] shows the internal structure of an LSTM node. x_t, h_t, $\tilde{C}_t$, and C_t represent the input, output, candidate cell state, and cell state at time t, respectively; f_t, i_t, and o_t represent the results of the forget gate, input gate, and output gate at time t, respectively. Their calculation formulas are as follows:

Forget gate: $f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$

Input gate: $i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$

Output gate: $o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$

Candidate cell state: $\tilde{C}_t = tanh(W_c x_t + U_c h_{t-1} + b_c)$

Cell state update: $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$

$h_t = o_t * tanh(C_t)$

Where W_f, W_i, W_c, W_o, U_f, U_i, U_c, U_o are weight matrices; b_f, b_i, b_c, b_o are bias vectors; and σ is the sigmoid function.

The forget gate controls what content to discard from the cell state. Information such as compensation amounts and timeframes in judgment results has no impact on the final label determination and is therefore forgotten during training. The input gate decides what new information needs to be added to the cell state. For example, in "dismiss [space] defendant [space] counterclaim [space] request," the word "defendant" is the object of "dismiss" and has a decisive impact on the final label, thus being updated into the cell state. The output gate controls the output content at the current time step based on the current input and cell state.

3 Experiments

3.1 Corpus Data and Evaluation Criteria

Currently, there is no standard test corpus for judgment result orientation analysis tasks. Therefore, the corpus data used in this paper was downloaded from China Judgments Online (http://wenshu.court.gov.cn/). Since civil case judgment results are typically not "one-sided"—often containing both supportive and non-supportive content for the plaintiff' s claims—this paper selected the three most common types of civil cases: contract disputes, tort liability disputes, and marriage/family disputes, with 500 judgment documents downloaded for each type. The final annotated dataset obtained after preprocessing is statistically summarized in .

Table 2: Annotated Dataset Statistics - Contract disputes: 1,433 supporting plaintiff (81.95%), 316 not supporting (18.05%), total 1,749 - Tort liability disputes: 346 supporting (71.17%), 140 not supporting (28.83%), total 486 - Marriage/family disputes: 470 supporting (80.57%), 114 not supporting (19.43%), total 584 - Overall: 2,249 supporting (77.92%), 570 not supporting (22.08%), total 2,819

The final experimental corpus consists of 2,749 labeled judgment results. This paper uses Accuracy (A) as the evaluation metric, where n_{right} and n_{wrong} represent the number of correct and incorrect predictions, respectively.

$$A = \frac{n_{right}}{n_{right} + n_{wrong}}$$

Four groups of experiments were designed: deep learning-based methods (including single-layer LSTM, two-layer LSTM, and three-layer LSTM) and dictionary-based rule methods (comparison experiment). In judgment results, since the object of judgment cannot be fixed—for example, "dismiss plaintiff" or "dismiss

defendant" —the specific object of "dismiss" must be determined to judge orientation. Therefore, unlike sentiment analysis which only considers sentiment vocabulary, the dictionary used in our comparison experiments consists of word collocations.

By continuously adjusting parameters to train the deep learning model until good results were achieved on the test set, the experimental results were compared with those of dictionary-based methods, as statistically summarized in . LSTM1, LSTM2, and LSTM3 represent single-layer, two-layer, and three-layer LSTM structures, respectively; P and N represent the two classification labels "support plaintiff" and "not support plaintiff."

Table 3: Experimental Results of Deep Learning and Dictionary Methods - For label P: LSTM1 97.8%, LSTM2 98.1%, LSTM3 98.5%, Dictionary method 94.2% - For label N: LSTM1 91.2%, LSTM2 92.3%, LSTM3 93.1%, Dictionary method 85.4% - Overall: LSTM1 96.4%, LSTM2 97.0%, LSTM3 97.7%, Dictionary method 92.1%

The results show that deep learning methods consistently outperform dictionary-based methods. The accuracy for the "support plaintiff" label is higher than for the "not support plaintiff" label, and increasing the number of LSTM layers improves accuracy.

[Figure 5: see original paper] shows the relationship between neural network layers and training time per iteration. Single-iteration training time increases linearly with the number of LSTM layers.

Table 4 shows the experimental results of single-layer LSTM versus dictionary methods across different datasets, where datasets C, TL, and WF represent contract disputes, tort liability disputes, and marriage/family disputes, respectively.

Table 4: Experimental Results of Single-Layer LSTM and Dictionary Methods Across Datasets - Contract disputes: LSTM 96.8% vs Dictionary 93.1% - Tort liability disputes: LSTM 95.7% vs Dictionary 91.2% - Marriage/family disputes: LSTM 96.1% vs Dictionary 91.8%

The experimental results reveal: a) For judgment documents of different civil case types, deep learning methods consistently outperform dictionary-based methods in both positive and negative classifications, with overall accuracy reaching up to 98.3%. This demonstrates that the proposed deep learning-based method has excellent scalability for judgment result orientation analysis tasks. b) Increasing neural network layers improves orientation analysis accuracy, but single-iteration training time grows linearly with network depth. Although the three-layer LSTM model improves accuracy by 0.7% over the single-layer LSTM model, training time increases by a factor of two. Therefore, in judgment result orientation analysis tasks, the single-layer LSTM deep learning neural network offers the best comprehensive performance. c) The accuracy for "not support plaintiff" labels is generally lower than for "support plaintiff" labels, mainly

because there is insufficient corpus of "not support plaintiff" type, preventing the deep learning model from fully learning various expressions in the corpus and resulting in lower prediction accuracy. d) Marriage/family disputes have the lowest accuracy for "not support plaintiff" judgments, primarily because orientation cannot be accurately predicted based solely on the semantics of the judgment result. For example, in a judgment result stating "The defendant is granted child custody," determining whether this supports the plaintiff requires knowing whether the plaintiff' s specific claim was for the defendant to have custody or for the plaintiff to have custody.

4 Conclusion

This paper introduces deep learning to judgment result orientation analysis for the first time, attempting to extract key features from unstructured texts, solve the problem of multi-entity recognition in judgment results using fuzzy matching methods, perform orientation determination through multi-layer LSTM-based deep neural networks, and construct an end-to-end orientation analysis model for judgment results. Experiments on judgment document datasets from three different case types achieved high accuracy, outperforming traditional dictionary-based rule methods and verifying the model' s good scalability and application value in judgment result orientation analysis tasks. This has important implications for future work on judgment document standardization and litigation lawyer recommendation.

The proposed model is currently limited to judgment result orientation analysis tasks. In general short-text sentiment classification, a single text may contain multiple viewpoints or sentiments, and whether this model is applicable requires further investigation. Additionally, some issues were identified during experiments. The fuzzy matching algorithm needs optimization, and some judgment results require more features for accurate orientation determination (for example, child custody judgments in marriage/family disputes require the plaintiff' s litigation claims as a feature for orientation analysis). Therefore, future work will focus on designing more sophisticated fuzzy matching algorithms and attempting to extract more features from judgment documents as model inputs to improve accuracy and stability.

References

[1] Liu X H. On the Publicity of Judgment Documents [D]. Chongqing: Southwest University of Political Science and Law, 2013. [2] Turney P D. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews [C]// Proc of Meeting on Association for Computational Linguistics. [S. l.]: Association for Computational Linguistics, 2002: 417-424. [3] Mohammad S M, Kiritchenko S, Zhu X. NRC-Canada: building the state-of-the-art in sentiment analysis of Tweets [J]. Computer Science, 2013. [4] Jijkoun V, Rijke M D, Weerkamp W. Generating focused topic-specific sentiment lexicons

[C]// Proc of Meeting of the Association for Computational Linguistics. [S. l.]: Association for Computational Linguistics, 2010: 585-594. [5] Pang B, Lee L, Vaithyanathan S. Thumbs up?: sentiment classification using machine learning techniques [C]// Proc of Conference on Empirical Methods in Natural Language Processing. [S. l.]: Association for Computational Linguistics, 2002: 79-86. [6] Wang S, Manning C D. Baselines and bigrams: simple, good sentiment and topic classification [C]// Proc of Meeting of the Association for Computational Linguistics: Short Papers. [S. l.]: Association for Computational Linguistics, 2012: 90-94. [7] Li S, Huang L, Wang J, et al. Semi-Stacking for Semi-supervised Sentiment Classification [C]// Proc of Meeting of the Association for Computational Linguistics and International Joint Conference on Natural Language Processing. 2015: 27-31. [8] Sun J W, Lv X Q, Zhang L H. Research on Chinese Weibo Sentiment Analysis Based on Dictionary and Machine Learning [J]. Computer Applications and Software, 2014, 31(7): 177-181. [9] Jiang J, Xia R. A Weibo Sentiment Classification Method Fusing Machine Learning and Semantic Rules [J]. Journal of Peking University: Natural Science Edition, 2017, 53(2): 247-254. [10] Graves A. Generating Sequences With Recurrent Neural Networks [J]. Computer Science, 2014. [11] Hochreiter S, Schmidhuber J. Long short-term memory [J]. Neural Computation, 1997, 9(8): 1735. [12] Cho K, Van Merrienboer B, Gulcehre C, et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation [J]. Computer Science, 2014. [13] Huang Z, Xu W, Yu K. Bidirectional LSTM-CRF Models for Sequence Tagging [J]. Computer Science, 2015. [14] Tang D, Wei F, Yang N, et al. Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification [C]// Proc of Meeting of the Association for Computational Linguistics. [S. l.]: Association for Computational Linguistics, 2014: 1555-1565. [15] Vo D T, Zhang Y. Target-dependent twitter sentiment classification with rich automatic features [C]// Proc of International Conference on Artificial Intelligence. [S. l.]: AAAI Press, 2015: 1347-1353. [16] Teng F, Zheng C M, Li W, et al. Multidimensional Topic Sentiment Orientation Analysis Model Based on Long Short-Term Memory [J]. Computer Applications, 2016, 36(8): 2252-2256. [17] Liang J, Chai Y M, Yuan H B, et al. Sentiment Analysis Based on Polarity Shift and LSTM Recurrent Network [J]. Journal of Chinese Information Processing, 2015, 29(5): 152-159. [18] Zhang C. Research on Text Classification Technology Based on Attention-Based LSTM Model [D]. Nanjing: Nanjing University, 2016. [19] Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult [J]. IEEE Trans on Neural Networks, 2002, 5(2): 157-166.

Note: Figure translations are in progress. See original paper for figures.