International Journal of Research in Signal Processing, Computing & Communication System Design
4 (2), 2018, 22-27
http://www.publishingindia.com/ijrsp

Classification, Information Extraction and Similarity Analysis of Indian Legal Cases

Aksheya Rajamani[1], Peeyusha Rathi[2], Richa Nagda[3], Utkarsha Nerkar[4*] and Mahesh Shirole[5]
[1]Department of Computer Engineering and Information Technology, Veermata Jijabai Technological Institute, Mumbai, Maharashtra, India. Email: arrajamani_b14@it.vjti.ac.in
2Department of Computer Engineering and Information Technology, Veermata Jijabai Technological Institute, Mumbai, Maharashtra, India. Email: pprathi_b14@it.vjti.ac.in
[3]Department of Computer Engineering and Information Technology, Veermata Jijabai Technological Institute, Mumbai, Maharashtra, India. Email: rjnagda_14@it.vjti.ac.in
[4]Department of Computer Engineering and Information Technology, Veermata Jijabai Technological Institute, Mumbai, Maharashtra, India. Email: ulnerkar_b14@it.vjti.ac.in
[5]Associate Professor, Department of Computer Engineering and Information Technology, Veermata Jijabai Technological Institute, Mumbai, Maharashtra, India.
Email: mrshirole@vjti.org.in
*Corresponding Author

Abstract: Computer technology can be useful in facilitating legal analysis in the law system. Lakhs of case files pertaining to Indian High Courts, over the past decade, are available in digital form. The problem faced by Indian lawyers and legal personnel is that they have to go through the routine of identifying the type of document and comparing relevant or similar cases. Currently, research has been done around the world to automate the process of text classification and information extraction of legal cases. Fully automatic or semi-automatic systems that carry out semantic text analysis are far less common. However, as per our knowledge, no research has been done to automate the tedium of document review process in India. In this paper, we aim to provide a hybrid approach by combining clustering and classification techniques and develop a system to automate this process in India, using Natural Language Processing and Machine Learning.

Keywords: Classifiers, Clusters, Features, Law system, Legal cases, Regular expressions, Similarity analysis.

I. Introduction

In a common law system, important information such as judgements passed, sections invoked, advocates for the plaintiffs, etc. aids legal personnel to frame strategies and take necessary decisions pertaining to a present case. In this era of digitalization, online access to the Indian legal cases generates opportunities and challenges for both the law practitioners and information technology researchers. As of December 2015, 38.70 lakh cases were pending across 24 high courts of the country [1]. The large digital database of legal cases has introduced a strong need for processing, analysing, retrieval of texts and presenting the information in a useful manner to the legal community. Though there are online platforms providing data related to Indian legal cases like 'IndianKanoon' [2], they lack the above-mentioned functionality.

Law firms expend significant time and resources on document review, a process requiring brief examination of thousands of documents. Therefore, a system that allows computers to take case texts as input, effectively classify cases, analyse similarity between different cases and present interpretations is extremely desirable. Not only will such a system save time for law practitioners, it can also potentially extract pertinent information that is not easily noticed by humans. Such a system is in much need because this tedious task of information extraction and reasoning from a massive amount of case descriptions will gradually expand beyond the capabilities of human beings. This system can thus be developed using technological advancements in the domains of computer science and machine learning.

Machine-learning oriented research in information extraction and document classification has been conducted and the search process has improved over the last two decades [10, 11, 12, 13]. But, because of the huge volume and the complexity of text data, finding relevant and required documents from the legal database without complete background knowledge is still a pretty difficult task. In the present scenario, there are tools that allow automatic recognition of the structural arrangement of law text and the textual browsing of cases. However, in India, not much research has been conducted in the domain of

legal document processing using Machine Learning algorithms. Wang and Nackoul [3] presented classification approach for predefined categories (murder, theft, rape, product liability and invasion of privacy) and Paul Thompson [4] considered some 40 broad high-level predefined categories. However, scalability and extensibility are limited for these approaches.

The proposed system uses Natural Language Processing techniques and Machine Learning algorithms to perform an extensive search through various legal cases in order to retrieve similar cases and extract information from them. The major contributions of this article are: i) extract features from raw data ii) detect relationships between documents and group them together using clustering iii) classify the input legal case iv) retrieve similar cases along with pertinent information.

The organization of the paper is as follows: Section 2 presents the related literature survey. Section 3 describes the proposed methodology and architecture of the system that we plan to implement. Section 4 presents the experimental setup of the proposed system and results of its implementation. Section 5 provides a brief conclusion of this article.

II. Related Work

This section discusses the contemporary research for the legal document processing, IR and machine learning approaches. A brief information is provided under the domain of clustering, classification and pre-processing of textual data. This section critically summarizes the current knowledge in the field of Document Classification using Machine Learning algorithms in Legal Domain.

The most commonly used vectorizer for the pre-processing involved in text classification is *Tf-Idf*. Ankit Basarkar [5] presented a comparative study of Binary, Count, *Tf-Idf* feature vectors and their impact on document classification. For each feature vector representation, they trained the Naïve Bayes classifier and then tested the generated classifier on test documents. In their results, they found that *Tf-Idf* should be the preferred vectorizer as it performed better than Binary and Count Vectorizer for document representation and classification.

The conventional data clustering approaches include partitional clustering algorithms such as K-Means and bisecting K-Means and agglomerative hierarchical clustering such as single-link, complete-link, average-link algorithms. The K-Means algorithm has been used by [6, 7, 8] for obtaining the clusters which led to good results. Conrad, *et al.* [9] discussed hierarchical clustering and soft clustering approaches for effective generation of taxonomies that can be used for classification.

The papers on Document Classification explore document representation, analysis of feature extraction methods and circumscribe different available classifiers such as KNN, SVM, Naive Bayes, Decision Trees and Neural Networks. Performance of different algorithms changes based on the data collection. However, SVM with term weighted representation scheme has shown some potential results in the tasks of text classification up to some extent. Basu, *et al.* [10] compare artificial neural network and support vector machine algorithms for use as text classifiers and the results show that the SVM algorithm is less complex than ANNs with a good performance. Sulea, *et al.* [11] presented that a system based on SVM ensembles can obtain high scores in predicting the law area and the ruling of a case, given the case description, and the time span of cases and rulings. In the research paper by Silvestro, *et al.* [12], the decision tree algorithm has been evaluated with KNN and Naive Bayes Classifiers to automate the text categorization process of law cases. The performance of decision tree was superior to that of Naive Bayes and KNN in the comparative evaluation. Boella, *et al.* [13] present a module that classifies law texts into different categories in accordance with labelled data and suggest reorganizations which reflect the true similarities and overlaps between domains. They used the well-known Support Vector Machines (SVM) for this task since it frequently achieves state-of-the-art accuracy levels. Alex Ratner [20] has presented a comparative study of Multinomial Naive Bayes, SVM, Logistic regression for legal document classification and observed that among these, SVM and Logistic Regression attain the best performance. Most research papers have used Linear SVM, Naïve Bayes, Logistic Regression in the field of text classification as stated above.

Although research has been done in text classification around the world, the above systems have not been implemented in the Indian legal domain, as per our knowledge. Thus, there is a need for implementing an efficient system that will automate the process of legal document review.

III. Proposed Methodology

The high-level architecture for the proposed system is shown in Fig. 1. The proposed system consists of the training and testing phase. In the training phase, the training data consisting of legal text documents is first pre-processed and then fed to the clustering model. The clusters generated as the output of the clustering model are then labelled manually into suitable classes. The legal text documents along with their classes are fed into the classifier model for training. In the testing phase, a new legal document given as an input by the user is classified by the trained classifier. The system retrieves documents similar to the input case and their apposite information.

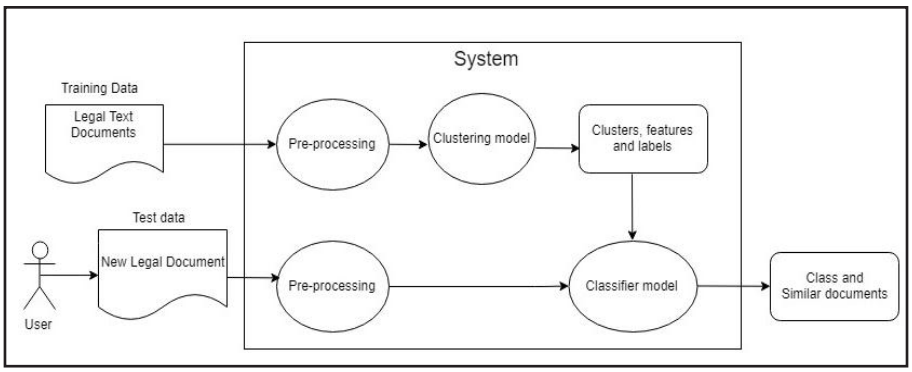

Fig. 1: High-Level Architecture for Clustering and Classification of the Legal Documents

A. Pre-Processing

In this stage, the corpus is formed by employing text extraction on our dataset of Indian criminal cases. The corpus is first tokenized and then lemmatized. The resulting corpus is then fed to the *Tf-Idf Vectorizer*. *Tf-Idf* [14], or term frequency-inverse document frequency, is a numerical statistic that determines the importance of a word to a document in a collection or corpus. The *tf-idf* value is directly proportional to the number of times a word appears in the document and is balanced by the frequency of the word in the corpus. There are words that occur many times in a document but may not be important. *Tf-idf* thus adjusts the weights of the words according to their rarity. The output of *Tf-Idf Vectorizer* is a feature matrix that we further use for clustering.

B. Clustering Model

We use hierarchical clustering approach to determine the number of classes, as we do not have prior knowledge of the classes present in the dataset. The number of clusters obtained by hierarchical clustering is used for applying k-means clustering to the same data and hand-tagged into different classes.

i. Hierarchical Clustering

The *tf-idf* feature matrix generated as the output of pre-processing is used to calculate a distance matrix using cosine similarity [15]. This distance matrix is fed to the linkage function for plotting the dendrogram. The dendrogram is then pruned to obtain the clusters of the corpus.

ii. K-Means Clustering

We use the number of clusters as '*k*' in the k-means algorithm [16] to verify if the clusters have been formed correctly. The inputs of the algorithm are the number of clusters *k* and the data set. The algorithm determines clusters based on the squared Euclidean distance between the data points and the centroids of the clusters.

The clusters formed using both hierarchical and partitioning approaches are compared to determine the final clusters formed. A set of top 20 feature words for every cluster is then used to label the clusters manually. These clusters are used to determine labels for classifying the test document.

C. Classification

The training data along with their appropriate labels are fed into the classifier model for training. When a new document is given as input by the user, the document is classified by the trained model into one of the categories that we get as a result of clustering. We use the following classification schemes for this purpose: Multinomial Naïve Bayes, Linear Support Vector Machine, Logistic Regression, Random Forest, Stochastic Gradient Descent Classifier [16]. The document class is determined based on the most probable predicted class by the above-mentioned classifiers, which means, if 4 out of 5 classifiers output class 'A' and 1 out of 5 gives an output of class 'B', then the class for that document is class 'A'.

D. Retrieval of Similar Cases and Information Extraction

The output that the user gets is the type/class of the test document and its similar cases present in the database. Once the class of the document is determined, we use cosine similarity to calculate the distance of the test document with every other document in that particular class. The documents that are similar to the test document are retrieved. This output also includes the following details pertaining to the case document using regular expressions: Judge(s), Advocate(s) for the appellant and respondent, the Court to which the appeal is made, Sections invoked in the case and judgement of the appeal.

The proposed system attempts to group items that are similar, on the basis of common characteristics. The similar case files retrieved and the information extracted from each case file is of key value to the user. This eliminates the need of browsing through the entire corpus of legal cases.

IV. Experimental Setup and Results

This section presents the dataset preparation and implementation of the proposed methodology. Scikit-learn [17], an open source machine learning library and a natural language toolkit has been used extensively for the same. The tabular and graphical representations of the obtained results are also highlighted.

A. Dataset Preparation

We have used the dataset of Indian Legal Cases from the website 'IndianKanoon' [2]. This website consists of all the case documents from different courts in India till date. The proposed system comprises of a dataset of around 800 criminal case files (PDF format) from the past ten years (2007-2017) belonging to 23 High Courts including Allahabad, Bombay, Gujarat, Delhi, Kerala, etc. These documents are criminal case appeals in the domain of murder, theft, rape, kidnap and dishonour of cheque. We split the dataset of eight hundred documents into a training set (720 documents) and cross-validation set (80 documents).

B. Information Extraction

We have extracted pertinent information from all documents - Judge(s), Advocate(s) for the appellant and respondent, the Court to which the appeal is made, Sections invoked in the case and judgement of the appeal using regular expressions. This was achieved by identifying patterns in the form of regular expressions, according to the common structure of most documents as shown in Fig. 2.

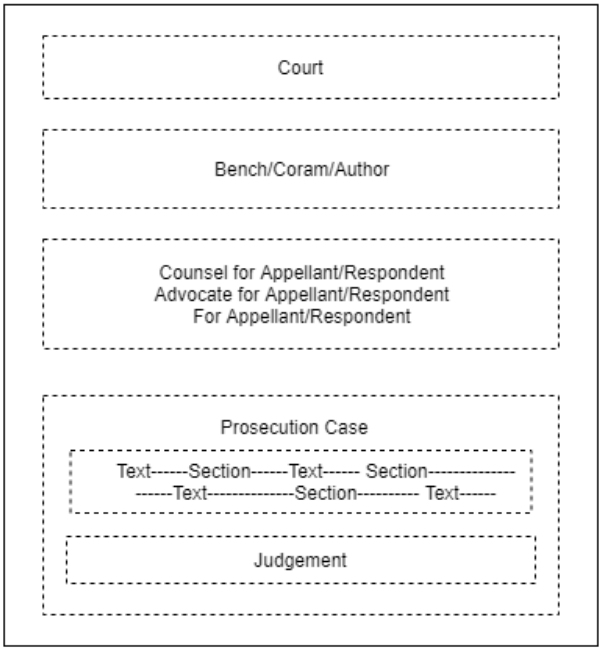

Fig. 2: Structure of Legal Document

C. Pre-Processing

A corpus of the documents is formed using *PyPDF2*, which is a pure-python PDF library used for splitting, merging together, and transforming the pages of PDF files into the required format. The corpus is first tokenized by removing "stop-words", punctuations and then lemmatized using *WordNetLemmatizer* [18]. We use *Tf-Idf Vectorizer* [14] to generate a feature matrix for which, best results were observed at maximum features limited to 1000, maximum document frequency at 30% and minimum document frequency to be 7 documents.

D.Clustering

Hierarchical clustering on the training set was performed to find the number of possible categories present in the dataset. In the agglomerative approach, we apply Ward's linkage method [19] to the distance matrix, the result of which, is used to plot the dendrogram shown in Fig. 3. The x-axis of the dendrogram represents the text documents and the y-axis measures the closeness of individual data points. The dendrogram is pruned at a height of 12, which results in 9 clusters. These clusters were tagged manually based on the top twenty features for every cluster. Following were the labels for the clusters: murder using weapon, murder by setting ablaze, acid attack, dishonour of cheque, kidnap for ransom, theft of electricity, theft of railway property and two classes under rape. We used 'k=9' for k-means clustering and observed that the clusters formed by both hierarchical ward clustering and k-means clustering were similar. The clusters so formed were used as classes for classifying the test document.

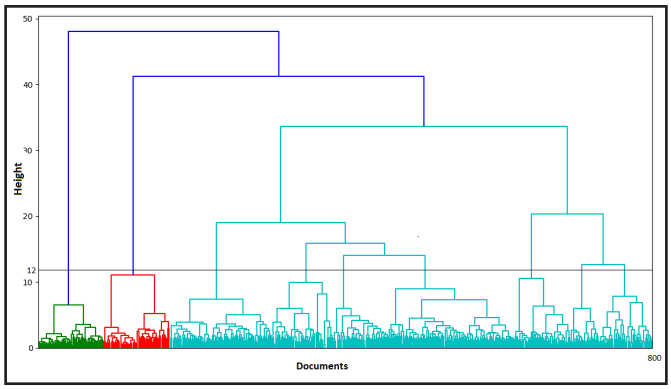

Fig. 3: Dendrogram Plotted as an Output of Ward Linkage Method in Hierarchical Clustering

E. Classification

We have executed Random Forest Classifier, a variant of Decision Tree, and Stochastic Gradient Descent Classifier and compared them with our implementations of the commonly used classifiers - Multinomial Naïve Bayes [12, 20], Logistic Regression [20] and Linear Support Vector Classifier [10, 11,

13, 20]. The confusion matrix of the implemented algorithms summarizes the prediction results on the classification model. It represents the number of correct and incorrect predictions with count values broken down by each class. The confusion matrix for Random Forest Classifier is shown in Table I and for Stochastic Gradient Descent Classifier is shown in Table II.

Table I: Confusion Matrix for Random Forest Classifier

Class	1	2	3	4	5	6	7	8	9
1	5	0	0	0	0	0	0	0	0
2	0	9	0	0	0	0	0	0	0
3	0	0	6	0	1	0	0	0	0
4	0	0	0	9	0	0	0	0	0
5	0	0	0	0	11	0	0	0	0
6	0	0	0	0	0	5	0	2	4
7	0	0	0	1	0	0	3	0	0
8	0	0	0	0	0	0	0	13	0
9	0	0	0	1	0	0	0	0	10

Table II: Confusion Matrix for SGD Classifier

Class	1	2	3	4	5	6	7	8	9
1	5	0	0	0	0	0	0	0	0
2	0	9	0	0	0	0	0	0	0
3	0	0	7	0	0	0	0	0	0
4	0	0	1	8	0	0	0	0	0
5	0	0	0	0	10	0	1	0	0
6	0	0	0	0	0	9	0	0	2
7	0	0	0	0	0	0	4	0	0
8	0	0	0	0	0	0	0	13	0
9	0	0	0	0	0	0	0	0	11

As shown in Table I, 9 out of 80 documents of the cross-validation data were misclassified during the implementation of Random Forest classifier. However, as per Table II, it was observed that only 4 out of 80 documents were misclassified by SGD classifier.

The accuracy, precision, recall, and f-measure of the classification algorithms applied to the dataset is as shown in Table III. Precision is a measure of exactness or quality i.e., a high precision indicates that more relevant results are returned by the algorithm than irrelevant ones. On the other hand, recall is a measure of completeness or quantity i.e., high recall indicates that most of the relevant results are returned by the algorithm.

Table III: Accuracy, Precision, Recall and F-Score Results for the Applied Classifiers

Classifiers	Accuracy	Precision	Recall	F-Score
Multinomial Naïve Bayes	0.87	0.89	0.81	0.83
Logistic Regression	0.95	0.94	0.94	0.94
Linear Support Vector Classifier	0.94	0.95	0.94	0.94
Random Forest	0.88	0.91	0.88	0.88
Support Vector Machine-Stochastic Gradient Descent	0.97	0.97	0.98	0.98

Due to the volume and complexity of textual data in our legal corpus, the number of irrelevant features is quite high. Random Forests are not able to manage large numbers of irrelevant features. This might be a reason why the accuracy of this classifier on our dataset is not as high as that of other classifiers, although it is comparable with Multinomial Naïve Bayes Classifier. On the other hand, Stochastic Gradient Descent is an iterative method which uses a different subset of the training data for each iteration. The randomness or noise introduced by the SGD algorithm helps to bypass the local minima to converge to a better minimum. We observed that the Random Forest Classifier achieved an accuracy of 88% when implemented using 350 trees. However, Stochastic Gradient Descent performs the best at 97% accuracy with 5 passes on the training data. SGD classifier achieved an improvement in accuracy of 2% over the other common algorithms implemented by us.

V. Conclusion

The process of legal document review using semi-supervised approach reduces the manual work to a great extent. The clusters of documents were formed and then categorized based on their domains and respective sub-domains. For example, the documents pertaining to the domain of murder were categorized as ‘murder using weapon’, ‘murder by setting ablaze’, ‘acid attacks’, those under theft as ‘theft of electricity’, ‘theft of railway property’, etc. Using these clusters as classes for further classifying the test document resulted in an accuracy of 97%. This model will thus, expedite the process of document review for legal personnel in the Indian law community. However, the constraint of this model is that, if a document which does not

belong to an existing class is given as input, it will still get classified into one of the classes, resulting in misclassification. This model can be scaled to incorporate all legal domains by training it on the entire Indian legal corpus. As a result, we can include all possible classes that any given document can potentially belong to.

References

[1] "Pending cases go down in Supreme Court, High Courts; but see upward swing in lower courts," *The Indian Express*, 01 October 2017. Available: www.indianexpress.com/article/india/pending-cases-go-down-in-supreme-court-high-courts-but-see-upward-swing-in-lower-courts-4869471/

[2] "Indian Kanoon - Search engine for Indian Law." Available: www.indiankanoon.org

[3] J. Wang, and D. Nackoul, "Classification and similarity analysis of legal cases," 6.863 Final Project, Spring 2010.

[4] P. Thompson, "Automatic categorization of case law," *Proceedings of the 8th International Conference on Artificial Intelligence and Law (ICAIL'01)*, pp. 70-77, St. Louis, Missouri, USA, 2001.

[5] A. Basarkar, "Document classification using machine learning," Master's Theses and Graduate Research, San Jose State University, Spring 2017.

[6] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," Technical Report #00-034, Department of Computer Science and Engineering, University of Minnesota, 2000.

[7] M. Chinea-Rios, G. Sanchis-Trilles, and F. Casacuberta, "Sentence clustering using continuous vector space representation," Pattern Recognition and Human Language Technologies Center, Universitat Polit`ecnica de Val`encia. (n.d.).

[8] U. Singh, and S. Hasan, "Survey paper on document classification and classifiers," *International Journal of Computer Science Trends and Technology (IJCST),* vol. 3, no. 2, pp. 83-87, March-April 2015.

[9] J. G. Conrad, K. Al-Kofahi, Y. Zhao, and G. Karypis, "Effective document clustering for large heterogeneous law firm collections," in *ICAIL'05*, ACM, Bologna, Italy, 6-11 June 2005.

[10] A. Basu, C. Walters, and M. Shepherd, "Support vector machines for text categorization," in *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03),* IEEE, 6-9 January 2003.

[11] O.-M. Sulea, M. Zampieri, S. Malmasi, M. Vela, L. P. Dinu, and J. van Genabith, "Exploring the use of text classification in the legal domain," in *2nd Workshop on Automated Semantic Analysis of Information in Legal Text (ASAIL'2017)*, London, United Kingdom, June 2017.

[12] L. D. Silvestro, D. Spampinato, and A. Torrisi, "Automatic classification of legal textual documents using C4.5," 2009. Available: http://www.ittig.cnr.it/

[13] G. Boella, L. D. Caro, and L. Humphreys, "Using classification to support legal knowledge engineers in the Eunomos legal document management system." (n.d.). Available: http://citeseerx.ist.psu.edu/viewdoc/download? doi=10.1.1.592.5468&rep=rep1&type=pdf.

[14] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of Massive Datasets*, 2011.

[15] P. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*, Pearson Addison Wesley, 2005.

[16] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction,* Springer-Verlag, New York, 2008.

[17] Scikit-Learn: Machine Learning in Python - Scikit-Learn 0.16.1 Documentation. Available: www.scikit-learn.org/.

[18] Python Programming Tutorials. Available: www.python-programming.net/lemmatizing-nltk-tutorial/

[19] Distances between Clustering, Hierarchical Clustering, 36-350, Data Mining. Available: http://www.stat.cmu.edu/~cshalizi/350/lectures/08/lecture-08.pdf

[20] A. Ratner, "Leveraging document structure for better classification of complex legal documents," Stanford University, 353 Serra Mall, Palo Alto, CA. (n.d.).