iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** IV **Month of publication:** April 2023

DOI: https://doi.org/10.22214/ijraset.2023.50123

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue IV Apr 2023- Available at www.ijraset.com

Legal Document Analysis

Khushi Udaysingh Chouhan[1], Roshni Sanjay Jha[2], Nikita Pradeep Kumar Jha[3], Shaikh Insha Kamaluddin[4], Dr. Sujata Khedkar
Department of Computer Science, VESIT, Mumbai

Abstract: ***Text preprocessing is the most essential and foremost step for any Machine Learning model. The raw data needs to be cleaned and pre-processed to get better performance. It is the method to clean the data and makes it ready to feed the data to the model. Text classification is the heart of many software systems that involve text documents processing. The purpose of text classification is to classify the text documents automatically into two or many defined categories. In this paper ,various preprocessing and classification approaches are used such as NLP, Machine Learning, etc from patent documents.***
Keywords: Tokenization, OvR classifier, LSTM model, Ensemble, Tf-idf, N-gram, Text classification

I. INTRODUCTION

Patent is a part of intellectual property. Effective patent analysis may bring lots of benefits for the enterprise. One of the main patent mining tasks is patent classification. Text mining and classification has always been a crucial application and topic for research since the foundation of digital documents. Around 80% of all information available is unstructured. Today, with increased digitization we have to deal with numerous text documents daily and thus the need for text preprocessing and classification has become a necessity. Text data contains noise in innumerable forms like emotions, punctuations, text in different case. Text preprocessing is a method to clean the text data so that it can be used in the model to get better performance. Text classification is one of the fundamental tasks in natural language processing (NLP) with broad applications. It is a machine learning technique that assigns a set of predefined categories to the text. The classifiers are used to organize, categorize and structure any kind of text document. Text classification can be done in two ways: manual or automatic.

Manual classification includes a human analyst who analyzes the content of the text and manually categorizes it. This may give better accuracy but is time consuming and strenuous.

Automatic text classification applies machine learning, natural language processing and other AI-guided techniques to classify the text in a faster and in more accurate way. For this project various preprocessing techniques have been implemented such as removing punctuations, lower casing the text, removing stop words and removing whitespaces. After the text is cleaned the text is tokenized and converted into vector form and then implemented with different classification algorithms. Algorithms implemented include OVR (OnevsRest), Random Forest classification, n-gram model, ensemble model and LSTM (Long-Short Term Memory). Finally the analyzed data is visualized to understand the results and observations better.

II. DATASET AND RELATED WORK

For the proposed system Contract Understanding Atticus Dataset (CUAD) v1 dataset has been used. CUAD falls under the legal domain. It was created with dozens of legal experts from The Atticus Project and consists of over 13,000 annotations. The files in CUAD v1 include 1 CSV file, 1 SQuAD-style JSON file, 28 Excel files, 510 PDF files, and 510 TXT files.

Existing projects using CUAD v1 dataset have implemented contract summarization algorithms. And one of the existing projects has experimented with four pre-trained variants of BERT for the purpose of fine-tuning their models.

III. METHOD

For better performance and accuracy the proposed system has used various classification algorithms. All the algorithms that have been used are explained below:

A. Random Forest Classifier

It is a supervised machine learning algorithm which is used for classification. This algorithm constructs a decision tree and the most voted prediction results will be considered as final prediction. Random Forest classifiers are suitable for dealing with the high dimensional noisy data in text classification.

RF is an ensemble of decision tree algorithms. Ensemble simply means combining multiple modes. Ensemble uses two types of methods i.e Bagging and Boosting. Bagging,also known as Bootstrap Aggregation, is an ensemble technique used by random forests. It creates a different training subset from training data with replacement and the final output is based on majority voting.Many hyperparameters are used either to enhance the performance or to make the model faster. Some important hyperparameters are:

1) *n_estimators :* number of trees the algorithm builds before averaging the predictions.
2) *max_feaures :*maximum number of features random forest considers splitting a node.
3) *random_state:*controls randomness of the sample.The model will always produce the same results if it has a definite value of random state and if it has been given the same hyperparameters and the same training data.
4) *n_jobs:* it tells the engine how many processors it is allowed to use. If the value is 1, it can use only 1 processor but if the value is -1 there is no limit.

B. One-vs-Rest

This is the most commonly used strategy for multiclass classification. It is also referred to as One-vs-All Classifier or OvA. This involves splitting the multi-class dataset into multiple binary classification problem. Then a binary classifier is used to train on each binary classification problem. Then selecting the most confident and better model predictions are made.

The *scikit library* or *sklearn library* is the most useful and robust library for machine learning in python. This library provides a separate OnevsRestClassifier class that allows the one-vs-rest strategy to be used with any classifier. It is very easy to use and only requires a classifier, that is to be used for binary classification, that is passed as an argument to the *OneVsRestClassifier.*

After implementing and experimenting with various binary classifiers as an argument to OvR classifier, the proposed system has got the highest accuracy for the model using Logistic Regression class.

C. N-gram Model

N-grams are continuous sequences of words or symbols or tokens in a document. In NLP n-gram is a continuous sequence of n items generated from the given dataset or sample of text where n can be any numbers like 1,2,3, etc. .

- *Unigram or 1-grams:* Unigram is a one word sequence. To generate 1-grams the value of n=1 is passed in the ngrams function of nltk. Unigram gives an accuracy of 84.967%.
- *Bigram or 2-grams:* Bigram is a two-word sequence. For generating 2-grams the value for n is given as 2 in grams function of NLTK. Bigram gives an accuracy of 94.11%.
- *Trigram or 3-grams:* Trigram is a three-word sequence. We put n=3 to generate trigrams or 3-grams. Trigram gave an accuracy of 66.66%.
- *Hybrid:* As the name suggests, hybrid is a combination of unigram, bigram and trigram and thus has a sequence of one, two and three words. The ngram_range() function is used to give range to generate hybrids. Hybrid of unigram and bigram gave an accuracy of 92.15% and hybrid of bigram and trigram gives accuracy of 96.07% and hybrid of unigram, bigram and trigram gives an accuracy of 89.54%.

D. Ensemble model

Ensemble learning is a powerful machine learning algorithm that is used across industries by data science experts. The beauty of ensemble learning learning techniques is that they combine the predictions of multiple machine learning models. There are several ensemble learning techniques.Some of them are bagging, boosting, stacking, blending,etc. The proposed system uses stacking and blending and the results can be compared below.

1) *Stacking:* It is an ensemble learning technique that uses predictions from multiple models(for eg.OvR,Decision Tree,etc) to build a new model. Below is the explanation of simple stacked ensemble:

- The train set is split into 10 parts.
- A base model(suppose Decision Tree) is fitted on 9 parts and predictions are made for the 10th part. This is done for each part of the train set.
- The base model is then fitted on the whole train dataset.
- Using this model, predictions are made on the test set.
- Now the same steps are repeated for another base model(suppose OvR) resulting in another set of predictions for the train set and test set.

- The predictions from the train set are used as features to build a new model.
- This model is used to make final predictions on the test prediction set.

2) *Blending:* Blending follows the same approach as stacking but uses only a holdout(validation) set from the train set to make predictions. In other words, unlike stacking,the predictions are made on the holdout set only.Below is the explanation of the blending process:

- The train set is split into training and validation sets.
- Model(s) are fitted on the training set.
- The predictions are made on the validation set and the test set.
- The validation set and its predictions are used as features to build a new model.
- This model is used to make final predictions on the test and meta-features.

Among these two techniques blending gives better accuracy than stacking for patent classification. Blending gives an accuracy of 99.40% whereas Stacking gives accuracy of 91.79%.

E. LSTM (Long-Short Term Memory)

LSTM is a deep learning model .They are the type of Recurrent Neural Network which have a good capability of memorizing patterns. It is widely used for Sequential data.Components of LSTM include three gates : *Forget Gate, Input and Output gate*

1) *Forget Gate:* This gate is responsible for discarding information which is not required.It selects the information which can pass through the Input gate.
2) *Input Gate:* After the removal of non essential information , input gate decides which information is more important for making predictions.
3) *Output Gate:* This is the final stage where it decides which information should it carry further.

In modeling, It uses a sequential model where it adds several layers using LSTM which consists of an input layer , hidden layer and output layer. Combination of LSTM and NLP gives accurate results for Text Classification as it can eliminate unused information and it can handle long term dependency which can reduce the cost.It gives the best result among all the implemented models (Random Forest, Ensemble and N-gram model) with highest accuracy.

IV. FLOW OF THE PROJECT

A. Preprocessing

Typically, most data for text classification are collected from the web, through newsgroups, bulletins boards, or broadcasts. They are multi-source, and thus have different formats, different preferred vocabularies and also significantly different writing styles even for documents in the same genre. Thus there is a need for text preprocessing to get rid of the noises in the text corpus and get better accuracy for the model.

Some crucial preprocessing steps include :

- ➔ Remove punctuations like @!#$()*
- ➔ Removing Stop words
- ➔ Lower casing
- ➔ Tokenization
- ➔ Remove white spaces

Stop words are a set of commonly used words (in this case) legal documents in English. Examples of stop words in English are “a”, “the”, “are”, “is”, “and”, etc. There is a need to eliminate stop words from the text corpus so that we can get the set of words that have more significance since the stop words carry less information.Words like corpus and Corpus mean the same but when not converted to lowercase those two are represented as two different words in the vector space model. Tokens are building blocks of Natural Language. Tokenization is a way of separating a piece of text into smaller units called tokens.

B. Vectorization

Vectorization is a classic approach of converting input data into vectors of real numbers which is the format that ML models support. After preprocessing the clean data is vectorized to a readable format for classification models. There are plenty of ways to perform vectorization.

Some of the ways that the proposed system uses are:

1) *TF-IDF:* TF-IDF or Term Frequency- Inverse Document Frequency, is a numerical statistic that's intended to reflect how important a word is to a document.TF can also be normalized frequency score.It can be calculated using following formula:

$$TF = \frac{Frequency\ of\ word\ in\ a\ document}{Total\ number\ of\ words\ in\ that\ document}$$

IDF can be calculated using the formula:

$$IDF = \log(\frac{Total\ number\ of\ document}{Documents\ containing\ word}$$

The final TF-IDF score comes out to be:

TF-IDF = TF * IDF

2) *Count Vectorization:* In this technique, a document term matrix is generated where each cell is the count corresponding to the news title indicating the number of times a word appears in a document,also known as the term frequency.
3) *Word2Vec:* It is a group of models which helps derive relations between a word and its contextual words. Two important models inside Word2Vec are:

- *Skip-gram model:* In this model, the system takes a center word and a window of context(neighbor) words and it tries to predict context words out to some window size for each center word. So,our model is going to define a Probability Distribution i.e. probability of a word appearing in context given a center word and vector representations are chosen to maximize the probability.
- *Continuous Bag of Words model:* In CBOW,we try to predict the center word by summing vectors of surrounding words.In abstract terms,this is opposite of skip-gram.

C. Classification

Classification is a supervised learning approach in which it categorizes the data and classifies it accordingly.It can be used to classify both structured and unstructured data.. Below is the table describing the classifiers that have been used for Text Classification in the proposed system.

Table 1. Performance comparison between 5 algorithms on CUAD v1 dataset

Classifier	Sub-classifiers / class used	Accuracy obtained	Advantages	Disadvantages
Random Forest		88.23%	It can handle large datasets with higher dimensionality. It is comparatively less impacted by noise.	As it uses tree data structure, it creates a lot of trees and thus uses more computational power and resources. It takes longer training time as compared to decision tree classifier.
N-grams			It encodes not only keywords but also word ordering automatically. Model is not biased by hand coded lists and is completely dependent on	It requires a considerable amount of training text to determine the parameters of the model. It is unsettling that it can only
Unigram		84.967%		
Bigram		94.11%		
Trigram		66.66%		

Unigram+Bigram Bigram Trigram Unigram Bigram Trigram		92.15% 89.54% 96.07%	real data. Learning features is relatively fast and easy.	interpret unseen instances with respect to learned training data.
OvR	LogisticRegression	98.0%	This method is suitable for multi-class classification.	A major disadvantage of this method is that many models have to be created.
Ensemble Blending Ensemble Stacking	LogisticRegression, KNeighborsClassifier, DecisionTreeClassifier, GaussianNB and OneVsRestClassifier(DecisionTreeClassifier()) DecisionTreeClassifier and OneVsRestClassifier(SVC())	99.40% 91.79%	Ensemble models have higher predictive accuracy compared to individual models. Different models can be combined to handle different types of data. Ensemble models are always less noisy and are more stable.	Ensembling is hard to learn and any wrong selection of models may result in lesser accuracy than the individual models. Ensembling stacking method takes a longer time to execute. Ensembling models are expensive both in terms of space and time.
LSTM		99.95%	It can handle long term dependency and it can eliminate unused information which is not required for prediction.	It is easy to overfit and it requires more memory to train

D. Visualization

Data visualization is representation of data through use of graphics such as bar charts, line charts or plots.The visual display of information communicates complex data relationships in understandable form. It is an essential part of data analysis as it gives deeper insights into patterns depicted by visualization. Some of the tools that the proposed has used for Data visualization are :

- *Word Cloud:* It is a pictorial representation made of words which resembles a cloudy shape . It shows words with high frequency.
- *Texthero:* Texthero is a python text processing toolkit which is used to clean, analyze and visualize the data.Using Texthero

The text can also be visualized by finding *top words* or displaying scatter plots.

Fig4.4.1 Flow of the proposed system

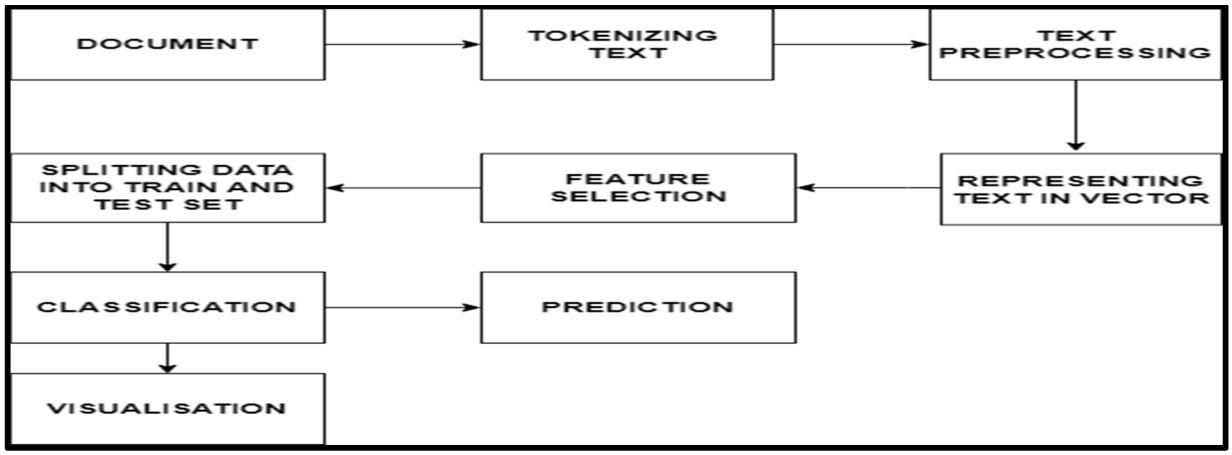


Fig 4.4.2 Visualization of words with the number of times they've appeared in the corpus.

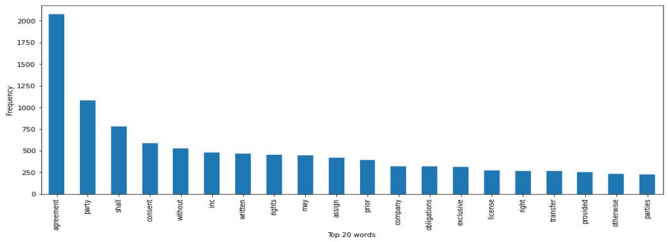
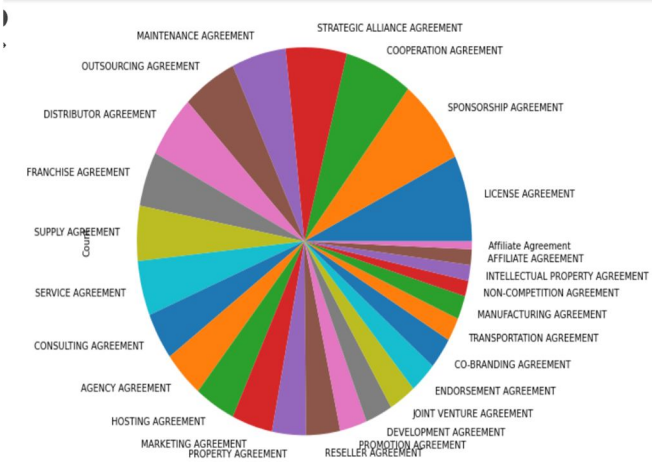


Fig 4.4.3 Wordcloud of the most frequent words

Fig 4.4.4 Pie chart showing the count of the type of contracts in the dataset



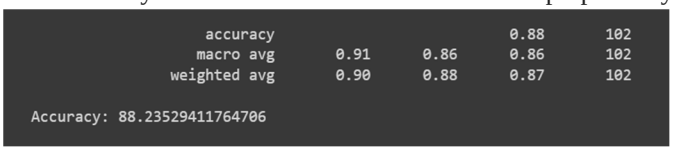
V. RESULTS

The proposed system has used 5 methods mentioned above. The results of these methods are:

A. Random Forest

This algorithm gave an accuracy of ***88.23%*** .

Fig 5.1.1 Accuracy of Random Forest Classifier for the proposed system



accuracy			0.88	102
macro avg	0.91	0.86	0.86	102
weighted avg	0.90	0.88	0.87	102

Accuracy: 88.23529411764706

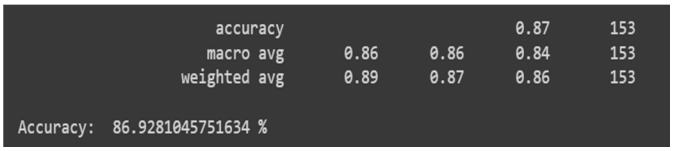
B. N-grams

This method includes Unigram, bigram, trigram and hybrid combinations of the three.

1) Unigram

The Unigram model using Logistic Regression classifier gave an accuracy of **86.92%.**

Fig 5.2.2a Accuracy of unigram model for the proposed system



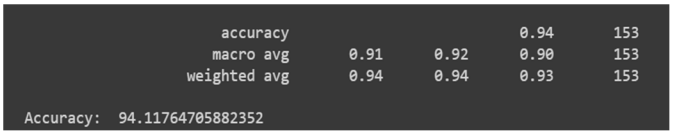
accuracy			0.87	153
macro avg	0.86	0.86	0.84	153
weighted avg	0.89	0.87	0.86	153

Accuracy: 86.9281045751634 %

2) Bigram

The Bigram model using the Logistic Regression Classifier gave an accuracy of **94.11%**.

Fig 5.2.3a Accuracy of unigram model for the proposed system



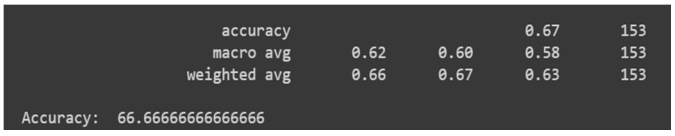
accuracy			0.94	153
macro avg	0.91	0.92	0.90	153
weighted avg	0.94	0.94	0.93	153

Accuracy: 94.11764705882352

3) Trigram

The Trigram model using Logistic Regression gave an accuracy of **66.66%.**

Fig 5.2.4a Accuracy of trigram model for the proposed system



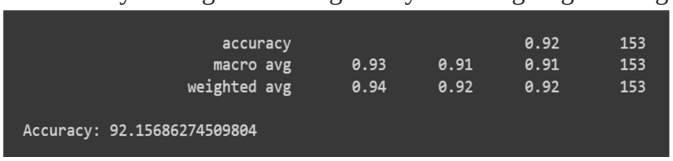
accuracy			0.67	153
macro avg	0.62	0.60	0.58	153
weighted avg	0.66	0.67	0.63	153

Accuracy: 66.66666666666666

4) Unigram and Bigram

The hybrid model of unigram and bigram gave an accuracy of **92.15%.**

Fig 5.2.5a Accuracy of unigram and bigram hybrid using Logistic Regression



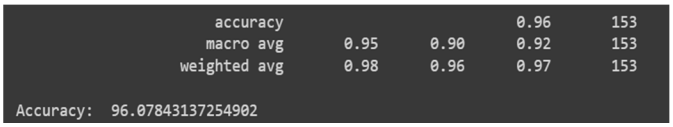
accuracy			0.92	153
macro avg	0.93	0.91	0.91	153
weighted avg	0.94	0.92	0.92	153

Accuracy: 92.15686274509804

5) Bigram and Trigram

The hybrid model of unigram and bigram gave an accuracy of

Fig 5.2.5a Accuracy of bigram and trigram hybrid using Logistic Regression



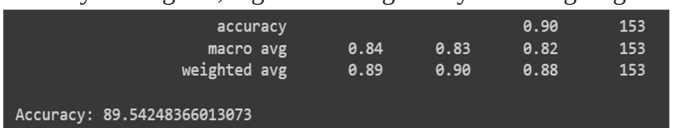
accuracy			0.96	153
macro avg	0.95	0.90	0.92	153
weighted avg	0.98	0.96	0.97	153

Accuracy: 96.07843137254902

6) Unigram, Bigram and Trigram

The hybrid model of unigram and bigram gave an accuracy of

Fig 5.2.5a Accuracy of unigram, bigram and trigram hybrid using Logistic Regression



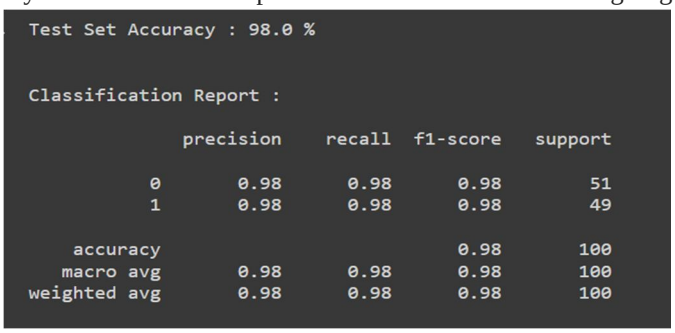
accuracy			0.90	153
macro avg	0.84	0.83	0.82	153
weighted avg	0.89	0.90	0.88	153

Accuracy: 89.54248366013073

C. OneVsRest

OvR multiclass strategy gave an accuracy of **98%.**

Fig 5.3.1 Accuracy and classification report of OneVsRestClassifier using Logistic Regression



Test Set Accuracy : 98.0 %

Classification Report :

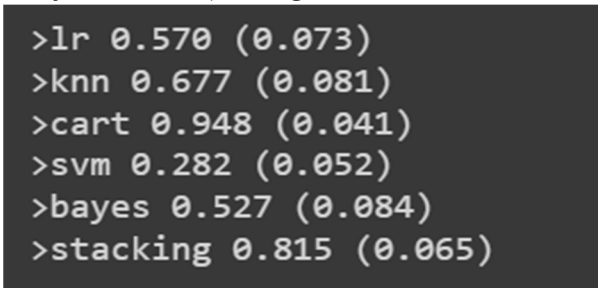
	precision	recall	f1-score	support
0	0.98	0.98	0.98	51
1	0.98	0.98	0.98	49
accuracy			0.98	100
macro avg	0.98	0.98	0.98	100
weighted avg	0.98	0.98	0.98	100

D. Ensemble

1) Stacking

Stacking of logistic regression, KNeighbours Classifier, Decision tree Classifier, Support Vector Machine Classifier and Naive Bayes Classifier gave an accuracy of 81.5%.

Fig 5.4.1a Stacking accuracy after stacking Logistic Regression, KNeighbours Classifier, Decision tree Classifier, Support Vector Machine Classifier and Naive Bayes Classifier.(showing accuracies in decimals without multiplication with 100)



```
>lr 0.570 (0.073)
>knn 0.677 (0.081)
>cart 0.948 (0.041)
>svm 0.282 (0.052)
>bayes 0.527 (0.084)
>stacking 0.815 (0.065)
```

Stacking of Decision tree Classifier and OneVsRest classifier gave an accuracy of **91.79%.**

Fig 5.4.1b Stacking accuracy after stacking Decision tree Classifier and OneVsRest Classifier.

```
>cart 94.524
>ovr 85.524
>stacking 91.790
```

2) Blending

After blending logistic regression, KNeighbours Classifier, Decision tree Classifier, Support Vector Machine Classifier, Naive Bayes Classifier and OneVsRest Classifier gave an accuracy of 99.408%.

Fig 5.4.2a Accuracy of the blending algorithm

```
    accuracy                           0.99       169
   macro avg       1.00      0.99      0.99       169
weighted avg       0.99      0.99      0.99       169

Accuracy: 99.40828402366864
```

Fig 5.4.2b Classification report of the blending model

	precision	recall	f1-score	support
0	1.00	1.00	1.00	4
1	1.00	1.00	1.00	4
2	1.00	1.00	1.00	8
3	1.00	1.00	1.00	4
4	1.00	1.00	1.00	8
5	1.00	1.00	1.00	8
6	1.00	1.00	1.00	13
7	1.00	1.00	1.00	5
8	1.00	1.00	1.00	4
9	1.00	1.00	1.00	7
10	1.00	1.00	1.00	5
11	1.00	1.00	1.00	12
12	1.00	1.00	1.00	6
13	1.00	1.00	1.00	11
14	1.00	0.88	0.93	8
15	1.00	1.00	1.00	3
16	1.00	1.00	1.00	5
17	1.00	1.00	1.00	1
18	1.00	1.00	1.00	5
19	1.00	1.00	1.00	3
20	1.00	1.00	1.00	10
21	1.00	1.00	1.00	10
22	1.00	1.00	1.00	13
23	0.89	1.00	0.94	8
24	1.00	1.00	1.00	4

E. LSTM

This deep learning model gave an accuracy of 99.95%.

Fig 5.5.1 Accuracy of LSTM for proposed system

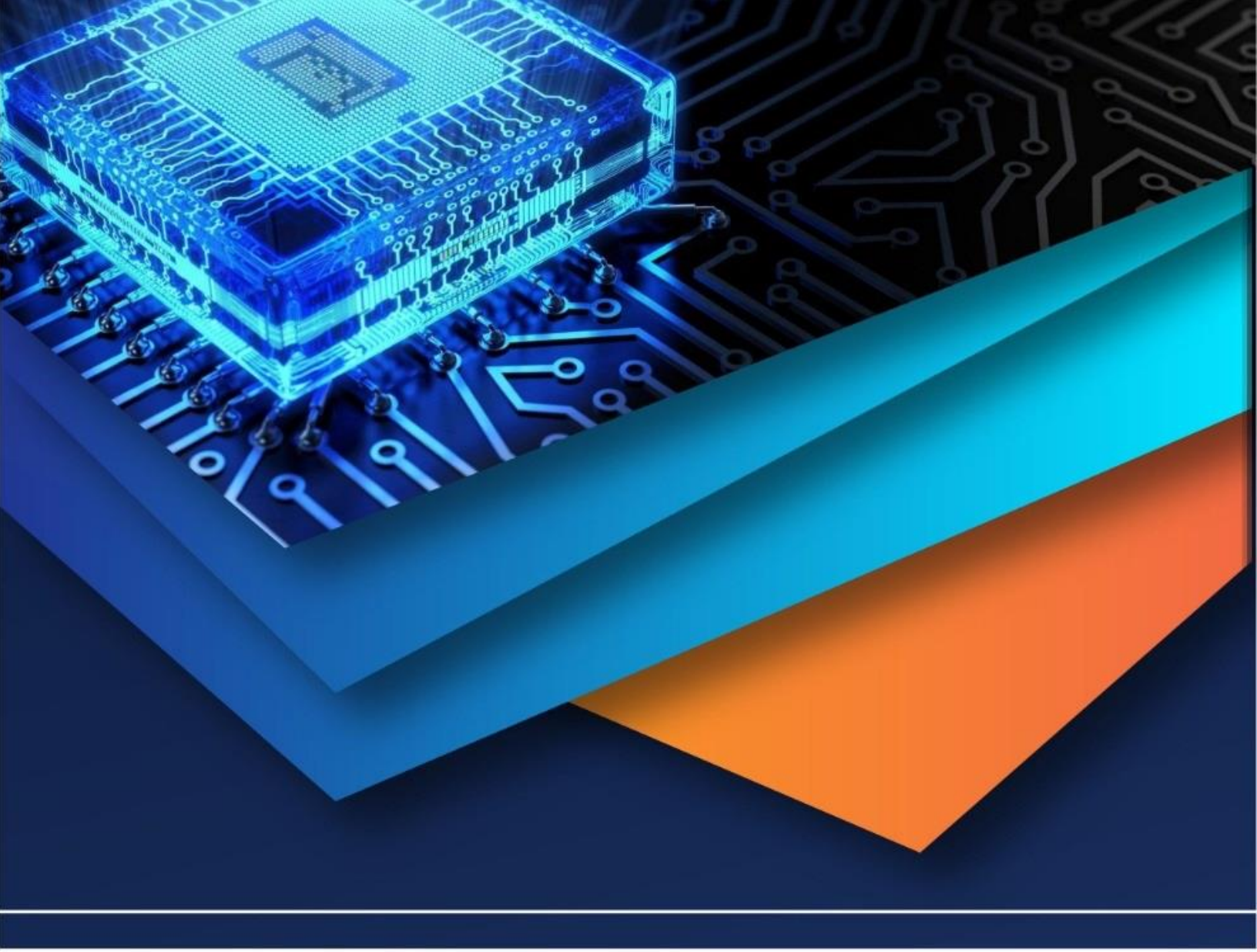
```
Accuracy: 99.95%
```

VI. CONCLUSION

Therefore, in this paper a high performance patent document analyzer and classifier is proposed and the results can be compared for various classifiers. The proposed system has implemented five methods for better accuracy and performance i.e. Random Forest Classifier, OvR model, N-gram model, Ensemble algorithm and LSTM(deep learning model). Thus a conclusion can be drawn that LSTM(Long-Short Term Memory) gives the best accuracy among all that is 99.95%. Also in future it is planned to explore many other techniques and also to try large datasets to improve the performance of our system.

REFERENCES

[1] Jie Hu, Shaobo Li ,Yong Yao, Liya Yu, Guanci Yang and Jianjun Hu, "Patent Keyword Extraction Algorithm Based on Distributed Representation for Patent Classification".
[2] Gaurika Tyag, "Ensemble Models for classification".
[3] Aishwarya Singh, "A comprehensive guide to Ensemble Learning(with Python codes)".
[4] Erin Yijie Zhang, "Legal Applications of Neural Word Embeddings".
[5] Kavitha Jayaram,Sangeeta K, "A Review: Information Extraction Techniques from research paper ".
[6] Susan Li, "Multi-Class Text Classification with LSTM".
[7] Nithyashree V, "What Are N-grams and How to implement them in python".
[8] Tom Lin, "Performance of Different Word Embeddings on Text Classification".
[9] Erin Yijie Zhang, "Legal Applications of Neural Word Embeddings".
[10] Charu Makhijani, "Advanced Ensemble Learning Techniques".
[11] Andreas Kanavos, Gerasimos Vonitsanos, Alaa Mohasseb, Phivos Mylonas, "An Entropy-based Evaluation for Sentiment Analysis of Stock Market Prices using Twitter Data".
[12] Maria Kalimeri, Vassilios Constantoudis, Constantinos Papadimitriou, Kostantinos Karamanos, Fotis K. Diakonos and Haris Papageorgiou3 , "Entropy analysis of word-length series of natural language texts: Effects of text language and genre"



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY