MultiCraft

International Journal of Engineering, Science and Technology
Vol. 13, No. 4, 2021, pp. 32-45

INTERNATIONAL JOURNAL OF ENGINEERING, SCIENCE AND TECHNOLOGY

www.ijest1-ng.com
www.ajol.info/index.php/ijest

An innovative approach to classify hierarchical remarks with multi-class using BERT and customized naïve bayes classifier

M.M. Dhina [1], S. Sumathi [2]*

[1,2*] *Department of Electrical and Electronics Engineering, PSG College of Technology, Coimbatore , INDIA*
**Corresponding Author, Email: ssi.eee@psgtech.ac.in, Tel.: +99947 59330*
ORCID iDs: http:/orcid.org/0000-0001-5165-0212 (Sumathi)

Abstract

Text classification is the process of grouping text into distinct categories. Text classifiers may automatically assess text input and allocate a set of pre-defined tags or categories depending on its content or a pre-trained model using Natural Language Processing (NLP), which actually is a subset of Machine Learning (ML). The notion of text categorization is becoming increasingly essential in enterprises since it helps firms to get ideas from facts and automate company operations, lowering manual labor and expenses. Linguistic Detectors (the technique of determining the language of a given document), Sentiment Analysis (the process of identifying whether a text is favorable or unfavorable about a given subject), Topic Detection (determining the theme or topic of a group of texts), and so on are common applications of text classification in industry. The nature of the dataset is Multi-class and multi-hierarchical, which means that the hierarchies are in multiple levels, each level of hierarchy is multiple class in nature. One of ML's most successful paradigms is supervised learning from which one can build a generalization model. Hence, a custom model is built, so that the model fits with the problem. Deep learning (DL), part of Artificial Intelligence (AI) , does functions that replicate the human brain's data processing capabilities in order to identify text or artifacts, translate languages, detect voice, draw conclusions and so on. Bidirectional Encoder Representations from Transformers (BERT), a Deep Learning Algorithm performs an extra-ordinary task in NLP text classification and results in high accuracy. Therefore, BERT is combined with the Custom Model developed and compared with the native algorithm to ensure the increase in accuracy rates.

Keywords: Machine Learning (ML), Multinomial Naïve Bayes (MNB), Natural Language Processing (NLP), Hierarchical classification, Artificial Intelligence (AI), Bidirectional Encoder Representations from Transformers (BERT), Deep Learning (DL)

DOI: http://dx.doi.org/10.4314/ijest.v13i4.4

Cite this article as:
Dhina M.M., Sumathi S. 2021. An innovative approach to classify hierarchical remarks with multi-class using BERT and customized naïve bayes classifier. *International Journal of Engineering, Science and Technology*, Vol. 13, No. 4, pp. 32-45. doi: 10.4314/ijest.v13i4.4

Received: November 10, 2021; Accepted: January 24, 2022; Final acceptance in revised form: February 7, 2022

1. Introduction

This paper introduces a method for text classification where the dataset is Hierarchical and Multiple-classes, by combining BERT and Customized Naïve Bayes classifier. This can be accomplished by studying Naïve Bayes Probabilistic Model, and BERT Model (Yu et al., 2019). The real world problems of NLP, is not just limited to one level of hierarchy. We could see many examples of such single level hierarchical problems such as: plain sentimental analysis, text summarization, detecting urgent issues, automating customer support processes and so on. This paper discusses the multiple levels of hierarchy, which we face in our world. Some merits with examples and comparisons are as follows:

1. In the blogging world, the traditional application of text classification is just limited to finding the title of the content (also called text summarization), but with the implementation of this paper it could be used to discover the sub-topics and also sub-topics of sub-topics. E.g., The base topic "Technology" has various sub-topics like Computer, Space, Nano and so on. Inside the sub-topic called Computer, we have another set of sub-topics such as: Quantum, AI, Cyber-Security and so on. There can be any number of hierarchies. In such real world problems, this paper solves the problem by providing the customized algorithm to process the different hierarchies.

2. In the customer support field, the complaint or issue raised by a customer should be modeled as multiple hierarchy instead of traditional single hierarchy. E.g., let's assume the complaint of a customer regarding a cloud product is: "The VM Machine of type ABC, has been down for X Days. This will impact my business a lot. Please provide support as soon as possible". The traditional way will be like: either detecting the complaint is urgent or not (also popularly known as Detecting urgent issues), or categorizing it into some problem category called VM Machine. If we had the model like, finding the urgency, followed by category and then detecting the problem duration could have a very great impact in the results (either customer satisfaction or sales growth) of the company, who provides the support.

The application could be unlimited with the hierarchical real world model and the paper completely focused on solving the multiple hierarchy dataset in a better way. Instead of having n ML classifiers to solve n hierarchy of a problem, the ML classifier in this paper is tuned to consider the results of the previous hierarchy. This paper also innovatively solves such kinds of problems discussed in the previous paragraph. The field of the problem is a manufacturing industry, the problem introduction, solution models and its impacts are discussed as follows:

The main problem in the Panel Board Manufacturing process or any Manufacturing process, is defects which may occur while manufacturing Panel boards. These defective panel boards undergo a testing process, which plays an important role in quality assurance (QA) by identifying the defects. The detected defects undergo re-processing in the industry. The testing process is manual and the remarks of the defected panel boards are generally written or noted (either through computer or hand written). Once the testing process is done the reprocessing employee's first objective is to classify the remarks into different sections, so that the defected panel board undergoes a certain re-processing.

The English text to explain the defect of the panel board may vary from employee to employee. Hence, the first step of the re-processing section utilizes intelligence of the employee to categorize the defect. Also, the time taken to categorize increases with increase in text. In ML, multiclass text classification is classifying each instance of text or data into one of three or more classes. Text classification into one of two classes is called binary text classification. Sentiment analysis to check whether the sentiment is positive or negative is a suitable example for Binary text classification. Multi-class text classification assumes that each instance is assigned to one and only one class. For example, a fruit can be either an apple or a pear but not both at the same time. In ML, Multi-label text classification is a variant in the classification problem. In Multi-label text classification, multiple labels are assigned to each instance. Mostly, Multi-class is confused with Multi-label classification, which is a generalization of multiclass classification. Multi-label classification is a single-label problem of categorizing samples into different classes, but an instance cannot be mapped to two classes whereas in the multi-label classification there are no constraints on how many classes an instance can be assigned. Every sample of the data in this paper can be classified to a particular class, therefore the paper discusses Multi-class text classification, not Multi-label classification.

If there is irrelevant and redundant data present in the dataset, then knowledge discovery during the training phase becomes a tedious process (Vidhya and Aghila, 2010). The First and foremost step for text classification is the pre-processing techniques such as Removal of Punctuation, removal of stop words, Lemmatization, Stemming and so on. It should be noted that Removal of stop words, Lemmatization and stemming is very much unfavorable to Deep Learning Classifiers. The reason is DL Classifier utilizes each stop word to considerable tokens and makes a meaningful sentence. The preprocessing technique is followed by Tokenization, Vectorization process, which is followed by training the classifier with the processed texts, then the Classifier can be tested with a new set of datasets. Many factors would affect the success of ML on a given task (Vidhya and Aghila, 2010), one such affecting factor of this problem is the hierarchy.

In a Panel Board Industry, the Panel board is tested for any defects, which may happen during the manufacturing process. Once the defects of the panel boards are detected, it is written in a sheet which may be stamped to the Panel Board or the defects are listed in a form through a computer corresponding to id of the product. In the Re-processing section of the industry, the Panel Boards defects which are stamped or listed are categorized into numbers (different classes). The classes are hierarchical in nature i.e., The sub-classes of each classes belong to one certain class and never intersect with other classes. The re-processing is a manual process which consumes days and hours of the employee. It should be noted that each writer of the panel board defect has a different vocabulary style for describing a defect. The employer also may make some errors in such cases and the main problem is the mapping of the remark to a number (Class) and its subclass. After this mapping process, the Panel Board undergoes Re-processing based on the classes, which was classified by the employer with the help of remark. The employer consumes more time, when the number of classes or types of defects are larger in nature. The time consumption may further increase if the number of level of hierarchy is higher in nature, each defect may even take an hour to detect the classes and its corresponding hierarchy. It is highly possible that a hierarchically structured set of categories might have a high impact on the classifiers, which are used and built (Krendzelak and Jakab, 2015). Deep learning is able to learn without human supervision, drawing from data which is unstructured and unlabeled. The BERT is a pre-training model based on DL, which has refreshed the best performance of 11 NLP missions (Liu et al., 2019). First and foremost step is importing or installing the components required for BERT and different variations of BERT are available based on requirements. Then, the dataset must be converted from comma separated values (.csv) to tab separated values (.tsv) file. The data is then undergoes masking and Next Sentence Prediction (NSP) section, followed by prediction of the data and performance analysis. The Masking and NSP is briefly explained in this paper. Due to rapid growth in ML and NLP, algorithms could be used to classify text, instead of utilizing employees with a lot of effort and cost. The advantage of implementing algorithms would be tremendous as the computational time for algorithms was much faster and with reduced cost.

Section 2 describes the literature survey made on BERT and MNB Classifiers followed by hierarchical classification. Section 3, which explores the Implementation proposed. Section 4 details the results and discussion of the proposed paper, which is followed by Conclusion, Acknowledgement and References.

2. Literature review

A lot of research has been done on the classifiers of NLP and categorization of hierarchical data. Reading comprehension, Question answering and Summarization are all some common industrial utilization of natural language processing problems that are often tackled using supervised learning with specialized datasets. These evidence points to a viable direction for developing language processing systems in which the classifier learns to accomplish the work from real demonstrations (Radford et al., 2019). Data cleansing, normalization, transformation, extraction of features and selection are all forms of data pre-processing (Vidhya and Aghila, 2010). The algorithms for each step of data pre-processing and best performances were presented by Kotsiantis et al. in 2007 through their paper. Data preprocessing is used in Text Mining (Vijayarani et al., 2015) to collect valuable and non-trivial information from unstructured information (Shathi et al., 2016). Kannan and Gurusamy (2014), researchers, examined the challenges of preprocessing methods such as Tokenization, Stop word elimination, and Stemming for text documents in 2014. The most frequent techniques to text classification do not take hierarchy into account, which clearly states that hierarchies play a very important role in the real world problem. The motivation of this paper on considering hierarchies and Gurusamy's study on hierarchies correctly converges at this point (Shathi et al., 2016). In their work released in 2016, the authors Angiani et al. investigate the reasons underlying the increase in accuracy, which gives a thorough assessment of each strategy. Several alternative preprocessing methods are displayed, and their accuracy is compared to the accuracy of the other methods in their study. If the test data meets the requirements of the class' rule, then the strongest rule is subsequently designated as the test data's class (Angiani et al., 2016). By increasing the strength of rule after constructing the training data, the confidence value from each text and category may be enhanced. It's all about finding the most powerful rule for each test data set that covers one class, according to Arusada et al. (2017) that detailed the procedures taken by classifiers during testing in 2017.

Natural Language Toolkit (NLTK) is a collection of open source software packages, lectures, and worksheets for language modeling (Loper and Bird, 2002). The design of NLTK is described in the NLTK article, which was released in 2002 by Loper and Bird, which report on how it has been employed well in classrooms with various mixtures of linguists and data science academics (Bird et al., 2008). The relevance of the Naive Bayes Machine Learning technique has been acknowledged, hence a research of text document categorization and statistical event models has been undertaken (Vidhya and Aghila, 2010). In 2010, the authors Vidhya and Agila examined the various Nave Bayes feature selection approaches with text document categorization metrics. MNB is a supervised learning method based on probability that is known for its theoretical and operational simplicity (Shathi et al., 2016). The authors Shathi et al. (2016) proposed an effective and efficient approach for categorizing text documents in order to give practical information retrieval utilizing the Naive Bayes algorithm in this research published in 2017. The independence assumptions that Naive Bayes classifiers are founded on nearly never hold for natural data sets, and particularly not for textual data, as has been observed. This idea has sparked three types of information extraction and machine learning research: 1) efforts to improve classifiers by loosening the independence assumption, 2) feature set tweaks to keep the independence assumption valid, and 3) tries to explain why the independence assumption is not necessary (Lewis, 1998; Qin et al., 2008), which describes a better observation on independence assumptions of MNB in a paper published in 1998 by Lewis.

Krendzelak and Jakab released a study in 2015 that included more research on text classification using hierarchical structures. The fundamental reason is that there hasn't been a substantial difference in performance metrics of hierarchical trials. In a hierarchical structure, conceptual hierarchy depicts the link between things or concepts (Krendzelak and Jakab, 2015). If the evaluation/classification objective has more sophisticated characteristics, a more elaborate conceptual hierarchy is required. Currently, the majority of conceptual hierarchies are built qualitatively. Choi and Shin (2010) released a study in 2010 that describes two novel quantitative ways to build a theoretical multi-level hierarchy.Pattern information from the stack of questionnaire survey data, which held hidden insights about the issue area, was revealed by this factor analysis. Considering questionnaire survey data as another method of knowledge elicitation for pattern discovery, which is a novel concept. The AHP (Analytic Hierarchy Process) is a multi-criteria judgment process that is regularly utilized Choi and Shin (2010).

The hierarchic design and assessment phases are two parts of the AHP's decision applications. For the establishment of hierarchies, experience and skill in the subject area are crucial (Vargas, 1990). These performance measures usually assume category independence and ignore documents misidentified into categories that are similar to or which are not too far from similarity with the right categories in the category tree, through which Vargas proposed a hierarchical categorization performance process in 1990.

In 2003, Sun et al. presented an outline of the hierarchical classification issue and its remedies. Although the frequently used flat categorization performance assessment methodology may be used for hierarchical classification, the links between categories in the category structure are not taken into consideration (Sun et al., 2003). An author may have many names due to name variants, and multiple writers may share the same name. The performance of document retrieval, online search, database integration, and author attribution may be harmed as a result of name ambiguity (Han et al., 2005). The authors, Han et al., offer the hierarchical naive Bayes mixture model in this 2005 publication, which is used in the suggested technique that will be applied and examined in section 3.

BERT (Bidirectional Encoder Representations from Transformers) has excelled in a number of language comprehension tasks (Sun et al., 2019). The generic language model BERT, which was pre-trained on cross-domain text corpora, Book Corpus, and Wikipedia (a large collection of text), achieves excellent performance on a few natural language processing tasks after fine-tuning in regression tasks. However, task-specific and domain-related information are still needed to improve the BERT model's performance, and the research outlines more extensive fine-tuning method assessments (Xu et al., 2019). BERT generally has two configurations with a lot of parameters: $BERT_{large}$: 24 layers, 1024 hidden dimensions, and 16 attention heads (found in transformer) with a total set of parameters of 340 Million, $BERT_{base}$: 12 layers, 768 hidden dimensions, and 12 attention heads (found in transformer) with a total set of parameters of 110 Million (Xu et al., 2019).

3. Implementation of the proposed system

Many common Machine learning techniques for text classification assume a simple collection of categories. As these classifiers are highly domain focused they can not be used for other kind of text classifications (Krendzelak and Jakab, 2015). There are totally 21 Custom_codes id, indicating they are Multiclass in nature, the Remarks need to be classified into one of these 21 classes. Once the first level of classification is over, the second level of hierarchy need to be classified.

The 21 classes of first level are subdivided into many subclasses. There are totally 79 subclasses in the dataset. Further, this subclass is classified into 254 Secondary subclasses. Hence, the scope of this paper is to classify the remarks for three levels of hierarchy, all in Multiple classification not Binary. In order to assist browsing and other interactive components of information retrieval, texts or documents are broadly categorized into hierarchies of subjects, including those managed by Yahoo! and the Open Directory paper (Toutanova et al., 2002). The analysis of the datasets are made with the help of libraries such as matplotlib and seaborn to graph plots, which visualizes the distribution of the data and it will be helpful to know whether the dataset is balanced or unbalanced. Fig. 1 represents the data distribution of remark counts with respect to classes.

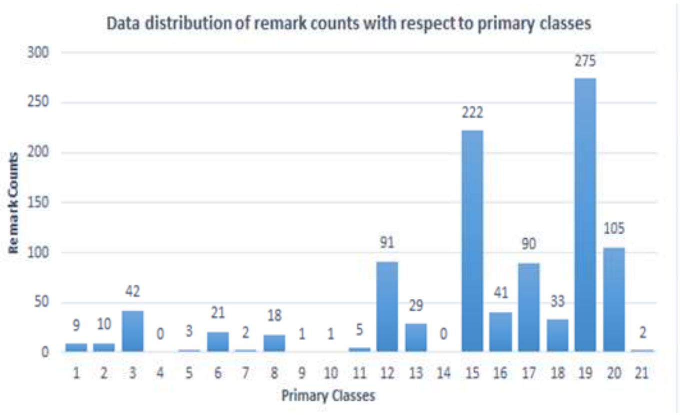
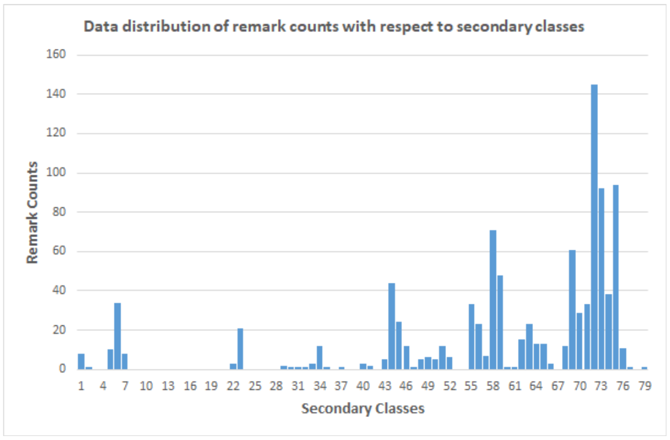
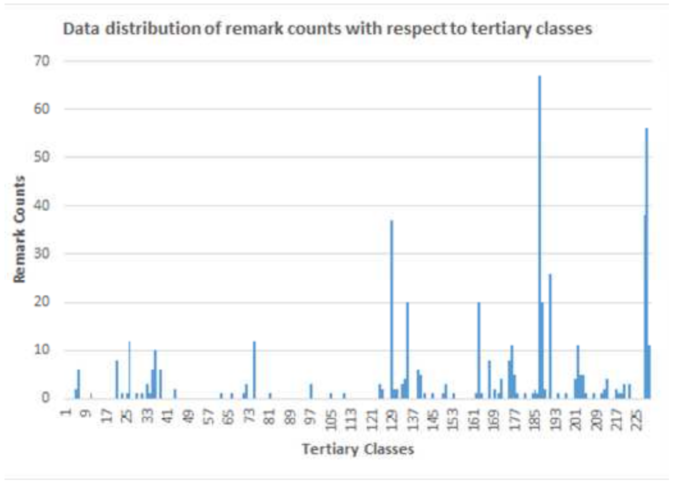


Fig 1 Data distribution of remarks count with respect to primary classes

Fig 1 shows the non-uniform data distribution of remark counts with respect to primary classes. Each dataset is a sample of 1000 rows and during the experiment, 3 such non-uniform datasets are available for training. The Efficiency of DL algorithms will be much better in higher test cases. Most of the small-scale industries have limited data compared to the large-scale (handling millions of data) industries. One more merit of this experiment is that even small-scale industries could get satisfactory results by implementing the innovative idea proposed in this paper. Fig. 2a represents the data distribution of remark counts with respect to secondary classes and 2b represents the tertiary classes, which are second and third level of hierarchy respectively.



(a)



(b)

Fig 2 Data distribution of remark counts with respect to (a) secondary classes and (b) tertiary classes

Fig.2a represents the second level of hierarchy and Fig. 2b represents the third level of hierarchy, where each class of the first level of hierarchy in Fig. 1 (parent class) may have zero to many secondary classes (child classes). The tertiary class, which is the third level of hierarchy also considered in the paper, which is shown in the Fig. 2b, representing the different classes in the hierarchy considered in this paper. To illustrate the further the Fig. 3 captures how the hierarchy looks like from the text to be classified.

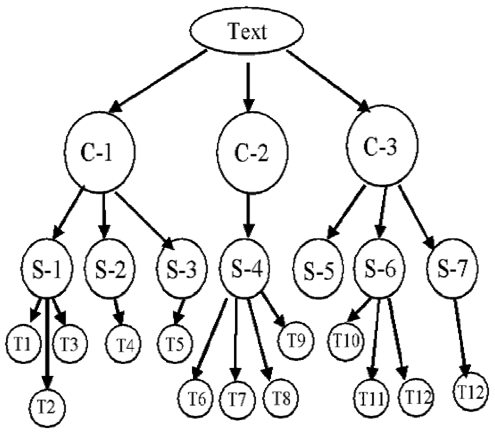


Fig.3 Three Levels of hierarchies to be classified from the text

Fig. 3 represents the different hierarchies that need to be classified, where C represents Classes, S represents the Secondary Classes and T represents Tertiary Classes. It must be clearly understood that each child has only one parent. E.g., Secondary class S-1 has only one primary parent class C-1. The text is classified into different classes, where each class is classified into different secondary classes and each secondary class may have zero to many tertiary classes. In the native Naïve Bayes, the errors didn't match, meaning the child class and parent classes didn't match. Hence, it became necessary to build the Naïve Bayes algorithm with this feature. The Naïve Bayes, is used for the second and third level of hierarchy alone. The first level of hierarchy is done by using BERT, a DL algorithm. Deep learning performances are better when the dataset becomes larger. In future technology, Deep learning will become the choice as statistical learning's performance may saturate at some point of level. But Deep Learning is not like that, as its performance keeps on increasing. BERT is an algorithm based on Recurrent Neural Network (RNN), which is based on Deep Learning. It doesn't come under statistical algorithms. The Fig. 4 shows the procedural flow of the Customized Naïve Bayes for hierarchical classification proposed in the methodology.

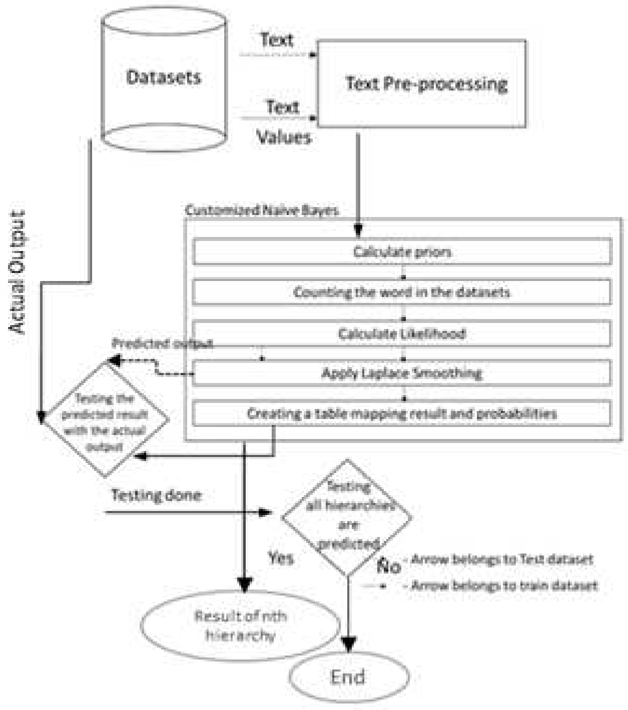

Datasets
Text
Text Values
Text Pre-processing
Customized Naive Bayes
Calculate priors
Counting the word in the datasets
Calculate Likelihood
Apply Laplace Smoothing
Creating a table mapping result and probabilities
Actual Output
Predicted output
Testing the predicted result with the actual output
Testing done
Testing all hierarchies are predicted
Yes
No
- Arrow belongs to Test dataset
- Arrow belongs to train dataset
Result of nth hierarchy
End

Fig 4 Block diagram for customized Naïve Bayes classifier

Fig. 4 shows the normal steps like data samples from dataset, preprocessing followed by customized steps of Naïve Bayes, which includes prior calculation, likelihood calculation, Laplace smoothing during Training. But only likelihood calculation and Laplace smoothing are needed for testing the data samples. Each block is briefly explained in this paper. The predicted outputs were compared with the actual outputs to find the accuracy and other metrics, followed by checking whether all three hierarchies are predicted or not. Once all hierarchies are classified, the process ends and the comparison of the results are done. The MNB classifier makes the strong assumption that the predictor variables are conditionally independent given the class (Eyheramendy et al., 2003).

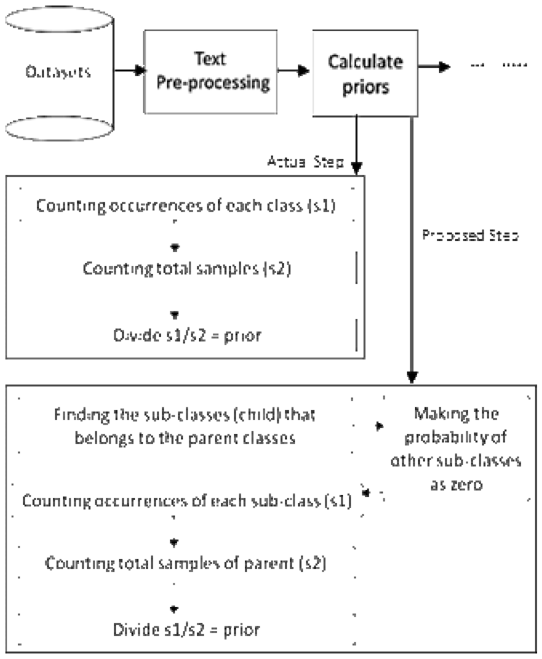

Datasets
Text Pre-processing
Calculate priors
... .
Actual Step
Counting occurrences of each class (s1)
Counting total samples (s2)
Divide s1/s2 = prior
Proposed Step
Finding the sub-classes (child) that belongs to the parent classes
Making the probability of other sub-classes as zero
Counting occurrences of each sub-class (s1)
Counting total samples of parent (s2)
Divide s1/s2 = prior

Fig 5 Block diagram for Change made in the customized MNB

The Figure 5 shows the actual change done in the general Naïve Bayes Classifier. During prior calculation, probability of each class in the data samples are calculated in the traditional methods. But in the proposed system, the probability of sub-class (child) which does not belong to the class (parent) is zero. Only the probability of the sub-class belonging to the parent class (result of the previous cycle of classifier) is calculated. Thus, the results are much better compared to the actual Naïve Bayes. This calculation tremendously impact the overall output and performance of the application. Additional advantage is the child-parent classes are properly matched, and reduced logical errors in hierarchical classification.

MNB model is simple to construct and is especially good for huge data sets and it is renowned to outperform even the most advanced categorizations due to its simplicity.

The following is the flow of the customized Naïve Bayes: i) Calculate Priors, ii) Calculate Likelihood, iii) Apply Laplace Smoothing, if required, iv) Create dictionary of words and its counts or probability, v) Train and test.

$$P(c/x) = \frac{P(x/c) \times P(c)}{P(x)} \tag{1}$$

In the Equation (1), the strong assumption of independence between classes is made. P(c|x) represents the probability of classes with respect to predictor words or posterior probability, which is need to be calculated. P(x|c) represents the likelihood probability and P(c) represents the class probability or prior probability and p(x) represents the predictor prior probability or marginal probability. Each feature can be multiplied to get the overall probability of a class in the given document X. The marginal probability is same for all classes, hence it need not to be considered.

$$P(c/x) = P(x_1/c) \times P(x_2/c) \times \ldots \times P(x_n/c) \times P(c) \tag{2}$$

The Equation (2) represents the probability of class c in the document X, where the probability of each word given the class is multiplied to get the overall probability of the class. The class having the higher probability is the result of the remark or text.

Probability of class (i) = (Number of counts with class i)/(Total number of all classes) (3)

The Equation (3) represents the prior probability or probability of class$_i$. Likewise the probability of classes is calculated. This is the general procedure of calculating prior in the native Naïve Bayes classifier. In the second level of hierarchy, the secondary class must be belonged to one parent class. The prior probability of other parent classes is made 0. So, the mismatching of child – parent relationship is retained. In the proposed methodology, the prior calculation is altered, which results in better accuracy.

The likelihood calculation for text classification is shown in Equation 4.

$$P(X_1 = x_1 \cap x_2 = x_2 \cap ... \cap x_k) = \frac{n!}{\prod_i x_i!} \prod_i p_i^{x_i} \tag{4}$$

The Equation (4) is used for discrete counts for a text classification problem. A multinomial distribution can be used to describe feature vectors in which each value indicates the number of occurrences of a phrase or its relative frequency, for example, If there are n elements in the feature vectors, each of them can take on k distinct values with likelihood p_k. Smoothing approaches consistently increase performance on a variety of length problems (Yuan et al., 2012). The frequency-based probability estimate will be 0 if a certain class and feature value never appears together in the training data. Because the probability estimate is related to the number of times a feature's value has occurred. This is troublesome because when the probabilities are compounded, it wipes away any information in the other probabilities. As a result, it's common to include a small-sample adjustment, known as pseudo count, in all probability estimates to ensure that no probability is ever set to 0. When the pseudo count is one, this method of regularizing Naive Bayes is termed Laplace smoothing, as shown in Equation 5.

.

$$P_r(j) = \log \pi_i + \sum_{i=1}^{|V|} \log(1 + f_i) \log(P_r(i \mid j)) \tag{5}$$

Equation (5), which represents the Lapalce smoothing solves the issue that, if a word appears again, the probability of it appearing again goes up. In order to smooth this, taking the log of the frequency makes the efficiency better. These 5 equations provide the basic knowledge about Naive Bayes Model. Understanding these 5 equations and its relation to text classification could help one to customize or re-tune the equation to solve unique problems.

In the BERT model, the first and foremost step is importing or installing the components required for BERT and different variations of BERT are available based on requirements. Then, the dataset must be converted from comma separated values (.csv) to tab separated values (.tsv) file. The data is then undergoes masking and Next Sentence Prediction (NSP) section, followed by prediction of the data and performance analysis. The Masking and NSP is briefly explained in this section. The flow of the BERT employed in this methodology is shown in the figure 6.

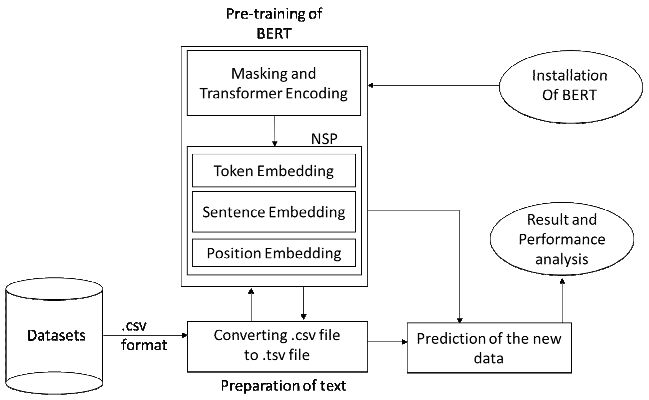


Fig 6 Block diagram for BERT procedure

The BERT procedure is depicted in Figure 6, in which the input word sequence is fed, and about fifteen percent of the words are progressively replaced with [MASK]. These [MASK] are then sent to the transformer encoder, along with a classification stage that assists in recognizing and substituting these [MASK] with key terms that are in context with the word sequence. The next phase in the process is to forecast the following phrase, which requires two tokens: [CLS] and [SEP]. Each of these tokens are used at specific times, with [CLS] being employed at the beginning of the text string and [SEP] being used at the end. The next sentence is detected using the Transformer design, which changes the output. The results from tokens are supplied to the classification layer, and the following sentence is predicted. The combination of the block diagram for 3 classifiers employed is shown in the figure 7, which also shows the classifiers employed for different hierarchies.

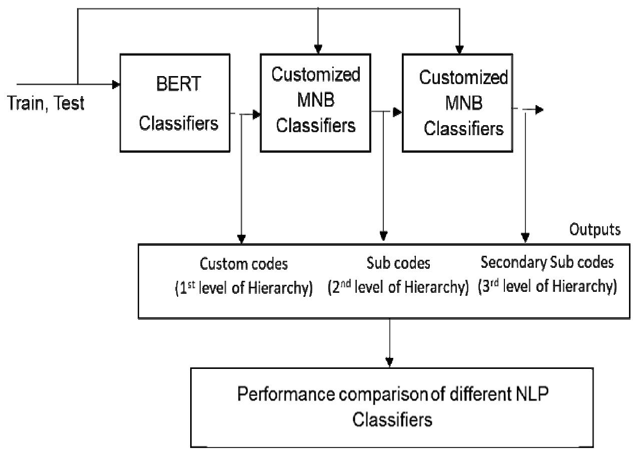

Train, Test
BERT Classifiers
Customized MNB Classifiers
Customized MNB Classifiers
Outputs
Custom codes (1st level of Hierarchy)
Sub codes (2nd level of Hierarchy)
Secondary Sub codes (3rd level of Hierarchy)
Performance comparison of different NLP Classifiers

Fig 7 Block Diagram of proposed hierarchical classifier

In the figure 7 BERT, a Deep Learning algorithm is employed to classify the first level of hierarchy. The results of first level automatically improve the further classes. Instead of Naïve Bayes, which performs better in low numbers of Data, BERT is employed for better performance. The second and third levels of hierarchies are classified by Customized Naive Bayes, which performs far better than actual Naive Bayes.

4. Results and discussions

A confusion matrix is a table that displays how a classification algorithm (or "classifier") performs on a testing dataset with known actual values. It visualizes how well an algorithm performs and allows for the rapid detection of class confusion. The performance measures such as Accuracy, Precision, Re-call, F1-measure, Kappa Score are computed from the confusion matrix. The True Positives, True Negatives, False Positives and False Negatives need to be calculated for each class. From the values of the confusion matrix, various metrics like Accuracy, Precision, Re-call, F1-Score, and its average along with the Kappa score are calculated using the library sklearn.

It could be noted that metrics recall or precision might reach a value of 1. But, to get the measurement that represents both Recall and precision, F1-Score is calculated. Because it penalizes extreme values, the F1-measure employs Harmonic Mean instead of Arithmetic Average. The F1-Measure will always be closer to the Precision or Recall number that is lower.

There are generally two of averaging:

i) Macro Average

ii) Weighted Average.

The Macro Average is the arithmetic mean of the metrics. The formula is given by Equation 6.

$$\text{Macro-Average} = (\text{Metric } 1 + \text{Metric } 2 + \dots \text{Metric } k)/k \qquad (6)$$

Where,

k represents the total number of samples.

In equation 6, n indicates the number of Metrics added. Macro averages only include the metrics, the weightage of the metrics have no significance in this equation. The macro average is calculated and shown in Table 1.

Table 1. Macro average of performance metrics between customized MNB and BERT + customized MNB

Macro Average		
Metric	Customized MNB	Customized MNB + BERT
Max. Precision (%)	68.1	64.6
Avg. Precision (%)	55.8	57.4
Min. Precision (%)	31.1	49.2
Max. Recall (%)	47.8	55.3
Avg. Recall (%)	35.4	52.3
Min. Recall (%)	16.4	47.1
Max. F1-Score	0.531	0.558
Avg. F1-Score	0.400	0.535
Min. F1-Score	0.186	0.464

From Table 1, it could be inferred that minimum maximum and average represents the macro average calculations from different datasets. The macro average represents average without considering the samples. Also, it could be noted that BERT + Customized MNB's average results are better compared to the Customized MNB.

The Weighted Average is the arithmetic mean, which includes the samples of each class results in weighted average, whereas in Macro average, the weights are equal. The formula is given in Equation (7).

$$\text{Weighted Average} = (D1*MetricDoc1 + D2*MetricDoc2 + \ldots + Dn*MetricDocn)/(D1 + D2 + \ldots + Dn) \quad (7)$$

In Equation 7, Dn indicates the number of samples of class n. The weighted average is tabulated in Table 2.

Table 2. Comparing weighted average of performance metrics between customized MNB and BERT + customized MNB

Weighted Average		
Metric	Customized MNB	Customized MNB + BERT
Max. Precision (%)	78.9	89.4
Avg. Precision (%)	73.9	84.9
Min. Precision (%)	65.0	81.7
Max. Recall (%)	78.4	90.7
Avg. Recall (%)	73.9	87.3
Min. Recall (%)	68.9	84.9
Max. F1-Score	0.762	0.897
Avg. F1-Score	0.712	0.854
Min. F1-Score	0.641	0.824

From Table 1 and 2, it could be understood that the weighted average sets weightage for each sample, which also helps to visualize the results clearly with the samples based on weightage. The Accuracy, Computational Time and Kappa Score, which is also a better performance metric of a classifier is shown in Table 3.

Table 3. Accuracy, Kappa score and time taken by customized MNB and BERT + customized MNB

Metrics	Customized MNB	BERT + Customized MNB
Max. Accuracy (%)	78.4	90.7
Avg. Accuracy (%)	73.9	87.63
Min. Accuracy (%)	68.9	84.9
Max. Kappa Score	0.687	0.856
Avg. Kappa Score	0.562	0.809
Min. Kappa Score	0.355	0.756
Max. Computational Time (s)	95.102	285.113
Avg. Computational Time (s)	92.537	263.343
Min. Computational Time (s)	89.509	249.918

From Table 3, when comparing Customized MNB and the BERT + Customized MNB, the Customized MNB's computational time is faster, for classifying all three hierarchies. The kappa score generally measures the agreement degree between the two evaluators, which also known as inter-rater reliability. In order to propose a classifier as best, it must be compared to other classifiers. Hence, the native MNB classifier is considered for the comparison of newly proposed classifiers such as Customized MNB and Hybrid classifier (BERT + Customized MNB). The comparison scenario is given as follows: the dataset is multiple hierarchy in nature, each hierarchy has multiple classes with non-uniform distribution and each class associated only with certain child-classes. The Customized MNB is compared with the Native MNB, with respect to statistical measures calculated from the accuracies. The Computation time for BERT is much higher, other than computation time all other metrics favors the hybrid algorithm. It could also be noted that the Kappa score and Accuracy are far better than the customized MNB. The reason for this tremendous change is explained later in this section. Other statistical parameters like median and Standard deviation are calculated and the calculated values are tabulated in Table 4.

Table 4. Statistical comparison of among MNB, customized MNB and BERT + customized MNB

Statistical Computations Of Accuracy	MNB	Customized MNB	BERT + Customized MNB
Mean (%)	57.8	73.96	87.63
Median (%)	54	74.6	87.3
Standard deviation (%)	17.5	3.90	2.37
Computation Time (sec)	12.11	92.53	263.3
Maximum Efficiency (%)	83.9	78.4	90.7
Minimum Efficiency (%)	34.1	68.9	84.9
Maximum Error Rate (%)	65.1	31.1	15.1
Minimum Error Rate (%)	16.1	21.6	9.3

From the Table 4, only few metrics like time, maximum efficiency and minimum error rate favors MNB, all the other metrics are in favor of Customized MNB indicating the performance are better in customized naïve Bayes. It could be noted that instead of maximum accuracy or efficiency, the minimum efficiency always considered as the important metric. Because, a classifier may even reach an efficiency of 90% or above due to over-fit or under-fit. Such classifiers may perform poor in real world problems, where each classification contains new sets of data, which completely differ from the training datasets. The Table 4 is graphically represented in Figure 8.

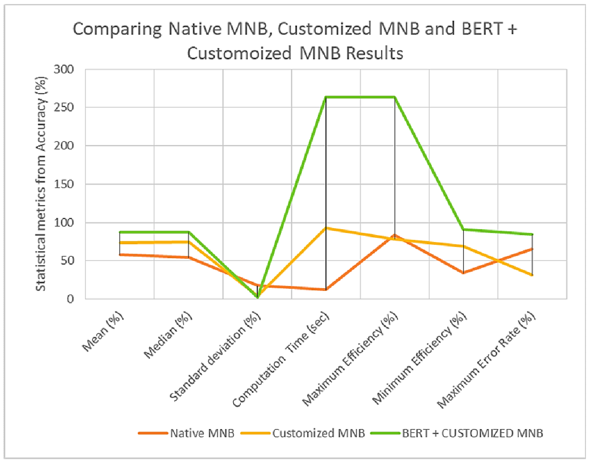


Fig. 8 Comparison of statistical metrics between Natie MNB and Customized MNB

In the Figure 8, it can be seen that mean and median are higher in customized MNB, whereas the standard deviation is lower indicates that the customized MNB is more consistent than Native MNB. The Minimum efficiency achieved by the customized MNB is much better than the native MNB. Maximum Efficiency could not be a very good metric, because in cases like over-fitting (the accuracy of the algorithm is more on trained datasets and the algorithm fitted mostly to the training datasets).

A better comparison among three classes between the two classifiers are compared and tabulated as Table 5.

Table 5. Accuracy comparison among MNB, customized MNB and BERT + customized MNB for different hierarchies

Classifiers	Accuracy/ Error Rate	Primary Class (First level of Hierarchy)	Secondary Class (Second level of Hierarchy)	Tertiary Class (Tertiary level of Hierarchy)
MNB	Accuracy (%)	57.8	40.2	41.3
	Error rate (%)	42.2	59.8	57.7
Customized MNB	Accuracy (%)	73.96	51.1	60.2
	Error rate (%)	26.04	48.9	39.8
BERT + Customized MNB	Accuracy (%)	86.73	64.63	72.03
	Error rate (%)	13.26	35.36	27.96

In Table 5, the accuracy of customized MNB is better than native MNB in all three hierarchies. Hence it could be said that customized Naïve Bayes wins over Native MNB. But the BERT + Customized MNB results are much better than the other two classifiers. The Table 5 is graphically represented for better visual comparison as shown in Figure 9.

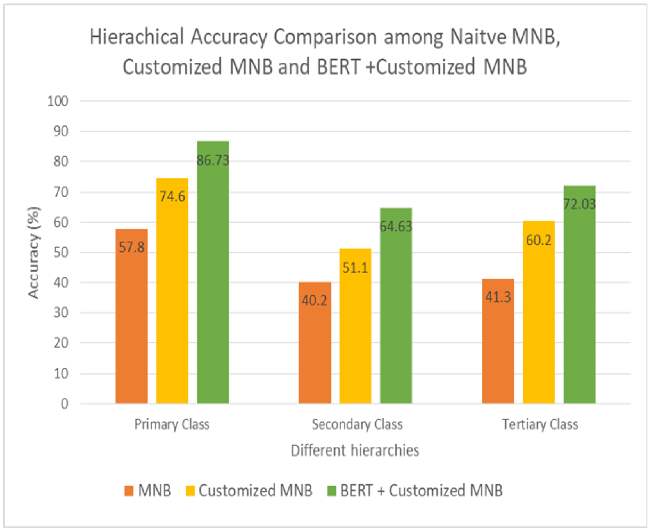


Fig. 9 Accuracy comparison of MNB, Customized MNB and BERT + Customized MNB for different hierarchies

From the Figure 9, it could be noted that the accuracy of customized MNB is higher in all three hierarchies. Another noticeable impact is reduction of error rate because of correct parent-child relation match, also a reason for accuracy rise. Finally, the Customized Naïve Bayes is taken as a ML and BERT included classifier as DL. The comparison of these two classifiers with respect to the number of data samples is graphed in the Figure 10. The reason for choosing the BERT + Customized MNB (hybrid Classifier) would be confirmed using the Figure 10. The comparison is done only for first level of hierarchy alone, as the influence of first level of hierarchy in overall performance is very higher.

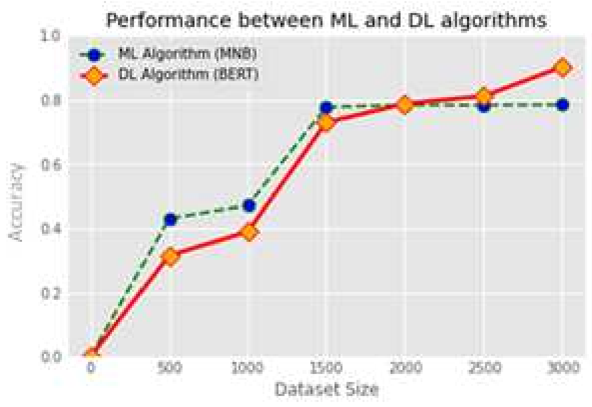
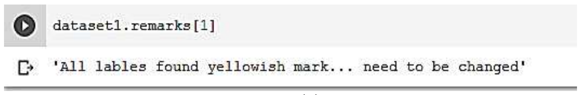


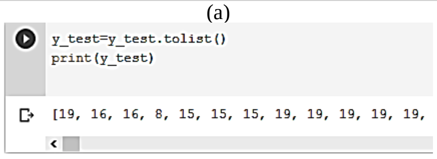
Fig 10 Comparison of two algorithms (DL and ML) with respect to number of Data Samples

From the Figure 10, it could be observed that Multinomial Naïve Bayes, which is a statistical Machine Learning Algorithm, performs better with a lower number of dataset samples. As the count of dataset samples increases, the performance almost gets saturated. If the Deep learning Algorithm is noticed, the performance is comparatively little lower at the early stage, but the performance keeps on increasing with the increase in the number of samples, which indicates that BERT + Customized Multinomial Naïve Bayes is becoming the right choice. So, a mix of classifiers yielded better results than utilizing any one type of classifier (Larkey and Croft, 2010).The sample inputs and outputs from the Google colabs is shown in the Figure 11.



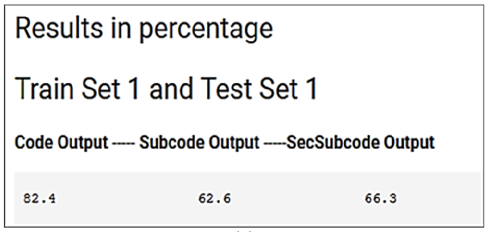
```
dataset1.remarks[1]
'All lables found yellowish mark... need to be changed'
```

(a)



```
y_test=y_test.tolist()
print(y_test)
[19, 16, 16, 8, 15, 15, 15, 19, 19, 19, 19, 19,
```

(b)


Results in percentage

Train Set 1 and Test Set 1

Code Output ---- Subcode Output ----SecSubcode Output

```
82.4                62.6                66.3
```

(c)

Fig. 11 Input (remarks) from dataset (a), Predicted output list (b), Overall result of three hierarchies saved as text (c) in Google Colabs

5. Conclusion

The text, which is the input for the classifier is the remarks of the defected Panel Boards found during testing in the panel board industry. The remarks of a defected Panel board are extracted from the dataset (any text format), which is converted to comma separated values (.csv) format and pre-processed in case of ML Classifier is used. For DL Classifier pre-processing techniques are not used, as pre-processing could lead to a lesser accuracy. The pre-processed data is then processed into tokens and vectors by techniques called tokenization and vectorization respectively, followed by training the classifier with the processed text. The ML classifier used is Native MNB at the beginning of the paper, the results are poor in the first level of hierarchy, as the level of hierarchy increases the accuracy of the classifier decreases. When classifying the second level of hierarchy, the errors in the class predictions are not logical, where the resultant child class did not match with the parent class (previous hierarchy) predicted. Hence the Native MNB is customized to avoid those logical errors. Multinomial Naïve Bayes is chosen to build from scratch to solve logical errors and it also met higher accuracy compared to the Native MNB. Then, customizing BERT is one such big effort and needs time, as a result Customized Naïve Bayes is employed for the second and third level of hierarchy, whereas the combination of customized Naïve Bayes with BERT met the goal of higher accuracy. One more advantage of DL classifier compared to ML Classifier is that the performance of the algorithm doesn't saturate like the ML algorithm. Finally, the algorithms are compared with various metrics and the best algorithm finalized in this methodology is the hybrid BERT + Customized Naïve Bayes Classifier, which met the primary goal of the client who is a Panel Board manufacturer and met the objectives of this methodology such as better accuracy, better computational time and reduced cost. The resources, software used in this methodology are open source in nature, and colabs offering high-processing speed are also free to use. Hence, the combination of BERT and customized Naïve Bayes is the conclusion of this paper.

Acknowledgment
The authors would like to thank Mr. Ian David, Business Partner, DJ Computing, Coimbatore for encouraging the paper providing resources and for his valuable ideas and support in consideration of this paper.

References

Angiani G., Ferrari L., Fontanini T., Fornacciari P., Iotti E., Magliani F., and Manicardi S., 2016, A comparison between preprocessing techniques for sentiment analysis in twitter, *Proceedings of the 2nd International Workshop on Knowledge Discovery on the WEB (KDWEB 2016)*, Cagliari, Vol. 1748, Sep. 2016.

Arusada M.D.N., Putri N.A.S., Alamsyah A., Training data optimization strategy for multiclass text classification, *IEEE 5th International Conference on Information and Communication Technology (ICoIC7)*, 17-19 May 2017, Melaka, Malaysia. https://doi.org/10.1109/ICoICT.2017.8074652

Bird S., Klein E., Loper E., Baldridge J., 2008, Multidisciplinary instruction with the natural language toolkit, Association for Computational Linguistics, *Proceedings of the Third Workshop on Issues in Teaching Computational Linguistics*, June 2008, Columbus, Ohio, USA, pp. 62–70.

Choi D.W., Shin J.G., 2010, Conceptual multi-level hierarchy for evaluation and classification, *IEEE 10th International Conference on Computer and Information Technology*, 29 June-1 July 2010, Bradford, UK. doi: 10.1109/CIT.2010.259

Eyheramendy S., Lewis D.D. and Madigan D., 2003, On the naive bayes model for text categorization, *9th International Workshop on Artificial Intelligence and Statistics*, pp. 332-339.

Kannan S., Gurusamy V., 2014, Preprocessing techniques for text mining, *RTRICS Multilingual Natural Language Processing Conference, Podi Conference*, October 2014

Kotsiantis S., Kanellopoulos D. and Pintelas P., 2007, Data preprocessing for supervised learning, *International Journal of Computer Science*, Vol. 1, pp. 4091-4096.

Krendzelak M., Jakab F., 2015, Text categorization with machine learning and hierarchical structures, *13th International Conference on Emerging eLearning Technologies and Applications (ICETA), 26-27 Nov. 2015, Stary Smokovec, Slovakia.* pp. 26-27, https://doi.org/10.1109/ICETA.2015.7558486.

Larkey L.S. & Croft W.B., 2010, Combining classifiers in text categorization, *19th ACM International Conference on Research and Development in Information Retrieval*, *Zu¨rich*, pp. 289–297.

Lewis, D.D. 1998. Naive (Bayes) at forty: The independence assumption in information retrieval. In: Nédellec, C., Rouveirol, C. (eds) Machine Learning: ECML-98. ECML 1998. *Lecture Notes in Computer Science*, Vol. 1398. Springer, Berlin, Heidelberg, pp.4-15, Aprill 1998. https://doi.org/10.1007/BFb0026666

Liu S., Tao H. & Feng S., 2019, Text classification research based on bert model and bayesian network, *IEEE Chinese Automation Congress (CAC)*, 22-24 Nov. 2019, Hangzhou, China. doi: 10.1109/CAC48633.2019.8996183

Loper E., Bird S. 2002, NLTK: The Natural Language Toolkit, *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pp. 63–70. https://doi.org/10.3115/1118108.1118117

Qin Y., Qing A., Wang X., Li X. and Liu W., 2008, Study on multi-subject text classification algorithm based on support vector machines, *Computer Engineering and Design*, Vol. 1, pp. 408-409.

Radford A., Wu J., Child R., Luan D., Amodei D. and Sutskever I., 2019, Language models are unsupervised multitask learners, *Open AI Blog*, Vol. 1, No. 8, pp. 1-24.

Vijayarani S., Ilamathi J. and Nithya, 2015, Preprocessing techniques for text mining-an overview, *International Journal of Computer Science & Communication Networks*, Vol. 5, pp. 7-16.

Vidhya K.A. and Aghila G., 2010, A survey of naive bayes machine learning approach in text document classification, *International Journal of Computer Science and Information Security*, Vol. 7, No. 2, arXiv preprint arXiv:1003.1795.

Shathi S.P., Hossain M.D., Nadim M., Riayadh S.G.R. and Sulthana T., 2016, Enhancing performance of naive bayes in text classification by introducing an extra weight using less number of training examples, *IEEE International Workshop on Computational Intelligence (IWCI)*, 12-13 Dec. 2016, Dhaka, Bangladesh, pp. 12-13. https://doi.org/10.1109/IWCI.2016.7860355.

Vargas L.G., 1990, An overview of the analytic hierarchy process and its application, *European Journal of Operational Research*, Vol. 48, pp. 2-8, 1990.

Yuan Q., Cong G. and Thalmann N.M., 2012, Enhancing naive bayes with various smoothing methods for short text classification, *21st World Wide Web Conference*, 2012. https://doi.org/10.1145/2187980.2188169

Han H., Xu W., Zha H., Giles C.L., 2005, A hierarchical naive bayes mixture model for name disambiguation in author citations, *ACM Symposium on Applied Computing*. https://doi.org/10.1145/1066677.1066920

Sun C., Qiu X., Xu Y., and Huang X., 2019, How to fine-tune BERT for text classification, *China National Conference on Chinese Computational Linguistics*, pp 194-206, May 2019. https://doi.org/10.48550/arXiv.1905.05583

Sun A., Lim E.-P., and Ng W.-K., 2003, *Journal of the American Society for Information Science and Technology*, Vol. 54, No. 11, pp. 1014-1028. doi: 10.1002/asi.10298

Toutanova K., Chen F., Popat K. & Hofmann T., 2002, Text classification in a hierarchical mixture model for small training sets, *10th International Conference on Information and Knowledge Management*, pp.105–112, Atlanta, Georgia.

Xu H., Liu B., Shu L., and Yu P.S., 2019, Bert post-training for review reading comprehension and aspect-based sentiment analysis, *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp 2324–2335, Jun. 2019. https://doi.org/10.48550/arXiv.1904.02232

Yu S., Su J., Luo D., 2019, Improving BERT-based text classification with auxiliary sentence and domain knowledge, *IEEE Access*, November 18.

Biographical notes

M.M. Dhina and Professor S. Sumathi are currently in the Department of Electrical and Electronics Engineering, PSG College of Technology, Coimbatore , India